časopis za informatiku i računarstvo

PERSONALNI računari 113

cena 16 din.




Naučno i Fantastično
Vodite računa o svom novcu.
Postoji razlika u kvalitetu koji
vam se nudi. Priuštite sebi najbolje
Nudimo vam standardnu OTC konfiguraciju
* "INTEL" 486DX4 "Overdrive™"/Pentium™ procesori
* Matične ploče "ASUS" (USA) 256/512 KB cache
* SIMM moduli sa parity generatorom ili EDO RAM
* Serijski portovi sa "BOCA" UART 16550C čipom
* "WESTERN DIGITAL" EIDE ili "IBM" SCSI tvrdi diskovi
* "HERKULES" ili "NUMBER NINE" PCI grafičke karte
* "FOCUS" klik tastature
* Mini Tower, Tower i File Server kućišta "MACASE"
OTC
COMPUTERS
SAVA CENTAR, BEOGRAD

olako potiskuje 486 procesor i u kućištima manje zahtevnih vlasnika računara. Šta se konkretno dobija ovom zamenom, saznajte na konkretnom primeru

SOFTVER

Boban Petković

Sve se češće čuju glasovi da je Windows 95 *u dobroj meri razočarao kupce. Sada se mnogi nalaze u nedoumici šta učiniti. Da li je* Warp *bolje rešenje?*

Dragan Grbić

Naš saradnik analizirao je rezultate zanimljivog ispitivanja koje su obavili američki stručnjaci: da li je *Microsoft Office* 95 bolji i intuitivniji od starog?

Srđan Pantić

Sve naše verzije Windows 95: Latinica, ćirilica po Microsoft-u

MACINTOSH

Milan Adamov

PowerTalk je deo sistemskog softvera koji omogućava efikasniji rad radnoj grupi korisnika Macintosh računara, obogaćujući sistemski softver mnogobrojnim kolaboracionim servisima

PROGRAMIRANJE

Saša Krajević

Kad neki proizvod koriste američko ministarstvo odbrane, Volvo, Krupp, Coca Cola i Pepsi Cola, treba li još bilo šta reći o njegovom kvalitetu? On se zove Progress

Dejan Vesić

Dobra vest za brojnu populaciju *Clipper* programera: najzad je izašla nova verzija. Oznaka je 5.3, ali to ne znači da je napredak u odnosu na 5.2 mali: jedna od novosti je i prelazak u *Windows* okruženje

računari 114

TEMA BROJA:
RAČUNARI U KUĆI

DODATAK:
INTERNET
mr Milan Milošević

Power Objects je novi alat firme *Oracle* za razvoj aplikacija nad bazom podataka sa minimalnim memorijskim zahtevima. Od *wizard*-a za kreiranje sesije do *drag-and drop* kreiranja formi i izveštaja, opisujemo kako OPO može olakšati vaš život.

Miljan Mitrović

Ma koliko se po strukturi i sintaksi razlikovali objektno orijentisani programski jezici, svi imaju nekoliko zajedničkih osobina. Mi ih nazivamo osnovama, a vi bez njih nećete ništa uraditi u svetu OOP-a

Ivan Sekulović

U današnje vreme grafičkih operativnih sistema, programer mora da pokaže i smisao za dizajn. Koja su pravila za kvalitetan izgled aplikacije, koji umnogome doprinosi njenoj upotrebljivosti?

Predrag Knežević

Na koji način izvesti komunikaciju mrežnih stanica? Jedno od mogućih rešenja je korišćenje nekog mrežnog protokola. Evo principa razmene poruka, funkcija IPX-a i njihovog korišćenja.

Žarko Berberski

U ovom nastavku upoznaćete nove ključne reči – namespace i using, koje je ISO C++ dodao radi lakšeg upravljanja velikim paketima klasa. Ovo je sveže proširenje koje još nije u sadašnjim verzijama uobičajenih prevodilaca.

Predrag Knežević

Uđite u samo srce sistemskog programiranja. Pozabavićemo se DMA kontrolerom koji može umnogome da ubrza pristup memoriji od strane periferijskih uređaja

MANIFESTACIJE

Goran Alimpić

Sredinom oktobra u Budimpešti je održan najveći sajam kompjutera u zemljama Istočne Evrope. Naš izveštač bio je na licu mesta i doneo mnoštvo utisaka

Vladan Aleksić

Da naša informatika ima zaista sjajnu perspektivu, pokazalo je 169 mladih stručnjaka koji su prezentovali svoje radove na ovogodišnjem simpozijumu SinFON održanom na Zlatiboru

Nenad Mitovski

Šta se dešavalo u hotelu *Beograd Intercontinental* na trećem okupljanju nazvanom *Microsys Computer Show*

Ivan Sajić

Šta se kod nas radi u oblasti neuroračunarstva? Pogledajte izveštaj sa ovog izuzetnog dvodnevnog seminara

RAZNO

Žarko Berberski

Zahvaljujući ljubaznosti nekoliko najvećih svetskih izdavača, u prilici smo da vam predstavimo najkvalitetnije svetske naslove koji se bave OOP-om u programskom jeziku C++

Ivan Gligin

Iako UNIX postaje prilično rasprostranjen kod nas, i dalje je puno onih koji su samo čuli za njega i postavljaju pitanje kakav je to sistem i koje su njegove mogućnosti, kakva je njegova istorija i rasprostranjenost, odnosno kakve veze ima ili nema sa MS-DOS sistemom. Ukoliko ste postavljali slična pitanja, a niste dobili odgovore, potražite ih u ovom članku

mr Milan Milošević

U poslovnoj praksi sve češće se koristi pojam elektronske razmene podataka (*Electronic Data Interchange*). Pogledajmo šta se krije za ovog naziva



NASLOVNA STRANA: Vladan Aleksić

Izdaje i štampa: BIGZ Bulevar Vojvode Mišića 17 11000 Beograd

Generalni direktor: Ilija Rapaić

Glavni i odgovorni urednik: Vladan Aleksić

Stručni urednik: Goran Akmpać

Urednici rubrika: Žarko Berberski, Jovan Bulajić, Vlada Čalić, Saša Krstić, Janko Stamenović, Ladislav Struhanik, Dejan Ristanović

Redakcija: Lada Muminagić, Igor Pilat, Darko Miletić

Stručni saradnici: Milan Adamov, Zoran D. Đirkić, Milan Čerić, Dragan Grbić, Dejan Jelović, Milan Mitrović, Ranka Jovanović, Milan Nikolić, Srđan Pantić, Boban Petković, Ivan Petrović, Ivan Sajić, Vladimir Stamenović

Urednik specijalnih izdanja: Vesna Ćosić

Tehnički urednik: Radoje Kavedžić

Grafičko oblikovanje: Miloš Ćupa

DTP: Tomica Orešković

Ilustrator: Milan Jovanović

Foto slog: Umetnički centar Beograd

Marketing: Aleksandra Banjac, Irena Ralić

Adresa redakcije: Bul. Vojvode Mišića 17/III 11000 Beograd

E mail adresa: *racunari@sezam.co.yu*

Telefoni
Redakcija, marketing: (011) 653 748
Agencija BIGZ-a: (011) 651 793, 653 565
Opšte poslovi pisma: (011) 652-403, 651 666/277
Poslovi prodaje i kontrole: (011) 650-894, 647 533
Pretplata: (011) 650-528, 651 666/226

Za dogovore i informacije o oglašavanju javite se na telefon 011/653 748, faks 011/648 140, modem 011/648-422

SEZAM
011 / 648-422 (15 linija)

Upravnik sistema: Vladan Aleksić (valex)

Sistem administratori: Dejan Ristanović (dejanr), Zoran Živolić (zzivolic)

Sekretarica: Jelena Blagojević

Zbog uspešnog pronalaženja uplata preko banke, molimo pretplatnike da nam posle svake uplate faksom ili pismom pošalju kopiju uplatnice

Pretplata

Za zemlju: Šest brojeva 77 din, jedanaest brojeva 140 din. Žiro račun D.P. BIGZ 40802 603 6 23264
Za inostranstvo: Šest brojeva 96 DEM, jedanaest brojeva 176 DEM. devizni račun D.P. BIGZ 40801 620 16101 820701 999 03377
Posebna doplata za avionsko slanje
Obavezna naznaka poziv na broj / šifra RG-06
žiro račun 40802 603 6 23264

Na osnovu mišljenja Ministarstva za nauku i tehnologiju Republike Srbije broj 413 00 198/95 01 od 07 juna 1995 godine časopis Računari je oslobođen plaćanja opšteg poreza na promet proizvoda kao publikacija od posebnog interesa za nauku

Pre izvesnog vremena susreo sam se sa poznanikom sa kojim nisam bio u kontaktu par mesec. On se ne bavi profesionalno računarima, čak ni ne poseduje svoj lični, ali redovno prati zapadnjačku štampu koja pokriva ovu oblast. Tema koja nam se vrlo brzo nametnula u razgovoru ticala se časopisa koji upravo imate ispred sebe - zamerka koja mi je upućena sa njegove strane odnosila se na slabu zastupljenost članaka posvećenih *Windows*-u 95. Svet je poludeo za najnovijim *Microsoft* ovim čedom, reče on, a vi još uvek pišete o DOS tehnikama i programiranju u asembleru!

Iskoristio sam pravo da ne budemo ludi, bio je moj odgovor. Najlakše je biti aktuelan - veliki broj knjižara nudi *Byte* i *PC magazine* samo koji dan po njihovom izlasku, a najnoviji softver nam je na domaku ruke. Međutim, prosečan čitalac "Računara" poseduje 386-icu sa 4 MB RAM-a - na njoj bez problema možete pokrenuti *Windows* 95, ali operativni sistem nije dovoljno pokrenuti, morate iskoristiti aplikacije koje rade pod njim. A to u pomenutoj kombinaciji nije nimalo lako. Stoga upotrebljivost stavljam na mesto ispred aktuelnosti - o novinama ćemo vas blagovremeno informisati, a na konkretnim stvarima će se bazirati uređivačka politika ovog časopisa.

Konkretna primena je i tema ovomesečnog broja, ovaj put u oblasti proizvodnje. Kakvu ulogu mogu tu imati računari? Ako zapitamo ekonomiste koji su glavni faktori za vrednovanje jednog preduzeća, dobićemo sledeći odgovor: nivo proizvodnje, produktivnost, zaposlenost, iskorišćenost kapaciteta, rast cena na malo, nivo zaliha sirovina i zalihe gotovih proizvoda. Uzmimo za primer *Mazdu* - poznati proizvođač automobila ima narudžbine za dve godine unapred, dok na lageru poseduje repromaterija dovoljan samo za dvodnevnu produkciju. Uzevši u obzir ovu činjenicu, kao i navedene faktore, nema dileme da li se u tom procesu koriste računari.

Pitanje je, da li se u *Mazdi* koristi *Windows* 95?

Vladan Aleksić

HARDVER

Skladištenje podataka

CD-ROM i izbrisivi optički disk u jednom uređaju



Dok čekate da se pojavi CD-ROM uređaj koji će po prihvatljivoj ceni omogućiti upis i očitavanje sa standardnog kompakt diska, razmislite o nabavci novog proizvoda britanske firme ImageStore. **ImageStore LF1004** kombinuje CD ROM uređaj četvorostruke brzine i izbrisivi optički disk kapaciteta 650 MB u jednom uređaju. Ideja ovog modela je da se omogući snimanje podataka na optički disk, da se nakon punjenja celog optičkog diska umetne novi a stari prekopira na kompakt disk i isprazni. Do sada su za slične mehanizme snimanja na kompakt diskove korišćeni hard diskovi velikog kapaciteta – ovaj hibridni model bi trebalo da otkloni većinu nedostataka korišćenja hard diska u ovom procesu. Jezgro uređaja se zasniva na *Panasonic PD* sistemu.

ImageStore LF1004 se isporučuje u eksternoj i internoj verziji: obe varijante se vide, ponašaju i priključuju kao standardni hard disk uređaji, a eksterne jedinice poseduju i vlastito napajanje. Novi hibridni uređaj odlično sarađuje sa operativnim sistemima *Windows for Workgroups, Windows 95, Windows NT, OS/2 Warp, DOS, Novell* i *Mac*. Cena uređaja je 599 britanskih funti.

Komunikacije

Štoperica za telefonske pozive

Ako imate probleme sa velikim telefonskim računima, švajcarski *Eurolink* možda ima rešenje za Vas. **Eltel phone manager** registruje sve telefonske pozive, kako one koje korisnik obavi, tako i one koje primi i snimi sve podatke o njima na PC. **Eltel** beleži datum i vreme poziva, dužinu razgovora, broj telefona koji je pozvan i obračunava cenu celog razgovora do 2000 telefonskih poziva. Registruju se i nepotpuni pozivi, a omogućeno je vođenje zasebne evidencije o poslovnim, internacionalnim i privatnim pozivima. Sam uređaj služi i za biranje specifikovanih brojeva i uspostavljanje veze. Pomoću ovog uređaja moguće je proceniti efikasnost telefonskih poziva, recimo odrediti procenat uspešnih poziva marketinških ekipa koje obave mnogo poziva pre nego što sklope posao ili odrediti koliko poziva otpada na privatnu, a koliko na poslovnu komunikaciju.

Multimedija

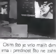
Mrežni pristup CD-ROM uređajima

Švedska firma *Axis Communication* proizvela je CD-ROM servere koji dozvoljavaju korisnicima da dele zajedničke biblioteke sa kompakt diskova. Uređaji se priključuju preko *Ethernet* ili *Token Ring* mreže i dozvoljavaju priključivanje do šest CD-ROM drajvova. Ovim se rešava problem korišćenja više baza podataka sa CD-ROM uređaja od strane više mrežnih korisnika. **Axis 850** se priključuje u *Ethernet*, a **Axis 950** u *Token Ring* mrežu. Oba modela poseduju integrisani fajl keš i postižu *data transfer rate* do 600 KB/s. Moguće ih je koristiti u *Windows for Workgroups, Windows NT, OS/2, NetWare* i *TCP/IP* okruženju. Oba modela su bazirana na 32-bitnim RISC čipovima, a podržavaju i SNMP. Cena obe verzije je 1665 nemačkih maraka.

Multimedija

Priključivanje personalnog računara na TV

Imagine Graphics iz Velike Britanije je na tržište izbacila proizvod koji bi trebalo da olakša multimedijalne prezentacije uz pomoć prenosnih računara koji imaju izuzetno slabe mogućnosti za efektan prikaz velikom broju posmatrača. Displeji *notebook* računara daju nekvalitetnu sliku, monitori od 17 inča su preglomazni za transport, a rešenje je u najrasprostranjenijem aparatima za prikaz slike, televizorima. Televizore je moguće naći i u tehnički slabo opremljenim objektima i vrlo brzo ih pomoću **Grand Media** PAL konvertora i *notebook* računara pretvoriti u osnovu multimedijalne prezentacije. Uređaj se priključuje na PC ili *Apple Macintosh* i služi za konvertovanje računarskog grafičkog VGA izlaza za prikazivanje na bilo kojem standardnom televizoru sa kompozitnim videom i S-VHS ili SCART konektorom. U pitanju je konvertor džepne veličine, prilagođenog nameni priključka koji se nosi zajedno sa portabl računarom na prezentacije koje mogu biti i van mesta stanovanja. Naravno, uređaj je vrlo koristan i u kombinaciji sa stonim računarom kada je potrebno održati prezentaciju pred većom grupom, a tada je veličina bitnija od kvaliteta slike koja se prikazuje i kada televizor predstavlja neuporedivo efikasnije sredstvo od monitora.



Osim što je vrlo malih dimenzija, adapter ima i prednost što ne zahteva poseban softver za rad. Uređaj koristi *frame buffer* i tehnologiju digitalnog video filtriranja da bi na ekranu televizora obezbedio stabilnu konvertovanu sliku bez "snega". Automatski se vrši prilagođavanje pozicije i kvaliteta slike u zavisnosti od ulazne frekvencije. Podržane frekvencije sa kojima adapter radi su od 24 kHz do 38 kHz horizontalno i 56 Hz do 72 Hz vertikalno, što adapter čini kompatibilnim sa grafičkim modom od 640 puta 480 tačaka. Takođe je moguće i prilagođavanje grafike od 800 puta 600 tačaka na televizijskom prijemniku sa standardom od 56 Hz. Cena aparata je 275 britanskih funti.

Prenosni računari

Mali, superizdržljivi snagator

Dell je proizveo novu **Latitude** liniju *laptop* računara, namenjenu korisnicima kojima ... elika snaga i dugotrajan rad



...enosnog računara. Ovaj projekat ...redstavlja značajan ...ovratak kompanije ...a tržište prenosnih ...oslovnih računara ...a kojem nije uče...tvovala nekoliko ...rethodnih godi...a. Nova *laptop* li...ija **Latitude XPi** ...e bazirana na no...im niskonapon...kim Intel Pentium čipovima na 75 i 90 MHz i poseduje izuzetno kvalitetne baterije visokog kapaciteta. **P90T** se isporučuje sa aktivnim matričnim TFT ekranom, 16 MB RAM i hard diskom od 810 MB. **XPi P90T** vrlo pametno i agresivno cedi baterije do kraja i time zaprepašćujuće produžava efektivno vreme rada bez zamene izvora napajanja. Štaviše, računar čak zbunjuje pojedine programe za testiranje baterija. Niskonaponski Pentium je Intel proizveo specijalno za prenosne računare koji traže uštedu u potrošnji struje svugde gde je to moguće. Sam čip radi na 3,3 V eksterno i

2.9 V interno. Zbog smanjenja napona rada smanjuje se zagrevanje procesora u odnosu na stari Pentium 90 čip, a samim tim i potrošnja energije. Posebnost novih proizvoda je baterija koja u kombinaciji sa štedljivim komponentama po dugotrajnosti nadmašuje sve konkurente. Računari se isporučuju sa litijum-jonskim baterijama visokog kapacita i produženog vremena pražnjenja. Očekuje se da će ovaj niskonaponski Pentium procesor uskoro suvereno zavladati tržištem prenosnih računara, ali se za sada pojavio samo u *Dell* modelima.

Sama **Dell XPi** linija prenosnika, na žalost, nije savršena. Modelima nedostaju neke osobine konkurentskih računara na koje su zahtevni korisnici najjačih prenosnika već navikli. Jedna od tih odsutnih osobina je i ugrađena podrška za zvuk. Sledeće razočaranje je što ekran podržava samo grafiku rezolucije 640 puta 480 tačaka. Bez obzira na ove nedostatke, **XPi** predstavlja savršeno rešenje za sve kojima dugotrajan ozbiljan posao na putovanjima zavisi od prenosnika. Osnovna konfiguracija sadrži Pentium 90 MHz čip, aktivni matrični TFT ekran, 8 MB RAM i disk od 340 MB i košta 4699 američkih dolara, a profesionalna verzija se isporučuje sa 16 MB RAM i diskom od 810 MB i košta 5398 dolara.

Izvori napajanja

Cink-vazdušne baterije koje pružaju 15 sati neprekidnog rada



Američka firma *AER Energy Resources* proizvela je **AER Energy PowerPro** eksterne baterije koje se priključuju ispod prenosnog računara i pokreću ga tokom 15 sati neprekidnog rada. **PowerPro** može da se koristi sa računarima iz serija **Toshiba Satelite T1900**, **Toshiba Satelite Pro T2400**, **T4700C**, **T4800CT** i **T4850CT**. Baterijski modul se montira ispod nekog od *Toshiba* računara i priključuje preko baterijskog ulaza. Ispravljač mrežne struje za *Toshiba* računare se koristi i za punjenje **PowerPro** baterijskog modula. Baterija je tako dizajnirana da po priključenju ostavlja slobodne sve prilaze drajvovima i portovima računara. **AER Energy PowerPro** prodaje se po ceni od 399 američkih dolara.

Zaštita opreme

Upravljanje napajanjem za štampače

Nightware je priključak za štampače koji se stara o tome da tokom neaktivnosti uređaji budu isključeni, a da im se napajanje automatski vrati kada se očekuje štampanje. **Nightware** registruje dotok informacija namenjenih štampaču i vraća napajanje štampaču, a s obzirom da štampač posle uključivanja nije odmah zagrejan i spreman za štampu, vrši i skladištenje pristiglih informacija i njihovo prosleđivanje tek kada je štampač spreman. Uređaj je proizvod američke firme *Micro Energetics* i košta 109.95 američkih dolara.

Video

Plug and Play MPEG kartica



Nova **9FX-PlusMPEG** kartica omogućava unos i pregled MPEG video fajlova bez potrebe da se menja *Windows* setovanje. Kartica radi na svim personalnim računarima opremljenim PCI veznicima. Osim što pruža visokokvalitetan video, kartica pruža i odličan sinhronizovani 16-bitni audio izlaz CD kvaliteta preko *mini-jack* priključka pomoću kojeg je moguće spojiti stereo zvučnike, eksterno pojačalo ili drugi PC preko njegove 16-bitne zvučne kartice i postići unakrsno miksovanje zvuka. Da bi se iskoristile prednosti i sve mogućnosti **9FX-PlusMPEG** kartice, neophodno je posedovati PC sistem opremljen nekim PCI grafičkim akceleratorom sa DCI podrškom. Cena **9FX-PlusMPEG** kartice je 199 američkih dolara.

Pomoćni pribor

Ultraolovka za brisanje

Nakon najezde raznih domišljatih uređaja za pokazivanje i biranje objekata, koji teže da potisnu klasičnog miša u drugi plan, pojavila se i paleta poboljšanih olovaka (pen uređaja). Neke od olovaka se odlikuju univerzalnošću, objedinjuju funkcije klasičnih miševa sa nekim dodatnim mogućnostima, a proizvod kanadske firme *Wacom Technology* pokušava da osvoji tržište usko specijalizovanim proizvodom **Erasing UltraPen**.

Ova olovka u zavisnosti od upotrebljenog pritiska briše više ili manje sa obojene površine - isto onako kao što funkcioniše i obična gumica za grafitne olovke. Olovka razlikuje 256 nivoa različitog pritiska, a u softveru koji nije kompatibilan sa ovim funkcijama brisanja, uređaj dozvoljava biranje i brisanje teksta ili ćelija jednim potezom. Cena **Erasing UltraPen** olovke je 89.99 američkih dolara.

Sledeći uređaji koje nudi isti proizvođač je **WideBody UltraPen With Pencil** koji košta 125 dolara i **DuoSwitch UltraPen** koji takođe košta 125 dolara. Poslednje dve olovke podržavaju standarde višestrukih tasterskih miševa u *Windows* i *Unix* radnom okruženju. *Wacom* je proizveo i nove grafičke komplete tablica koji sadrže **ArtPad II** i **Erasing UltraPen** za 174.99 dolara, **ArtZ II 6 * 8** i **Erasing UltraPen** po ceni od 389.99 dolara, **ArtZ II 12 * 12** i **Erasing UltraPen** po ceni od 539.99 dolara, **ArtZ II 12 * 18** i **Erasing UltraPen** za 869.99 dolara i visoko profesionalni **ArtZ II 18 * 25** i **Erasing UltraPen** koji košta 2449.99 dolara.

Zaštita opreme

UPS za file server računare



Britanski *Galatrec International* je proizveo uređaj za besprekidno napajanje **ByteBak UPS** kojim upravlja 16-bitni mikroprocesor proveravajući izvor struje 1600 puta u sekundi i obezbeđujući odgovarajuću struju kad god ima fluktuacije mrežnog napona. Sistem može da kompenzuje struju koju dobija čak i kada napon padne na 154 V. Ukoliko napon struje padne ispod 154 V u pogon se puštaju i baterije da bi ozbedile izlazni napon. Proizvođač tvrdi da ovakav način rada produžava vek baterija za 30 procenata. Na prednjoj strani uređaja nalazi se LCD koji prikazuje više od 20 različitih funkcija koje se prate, uključujući između ostalih i ulaz, nivo punjenja baterija i voltažu. Na displeju se prikazuju i poruke za alarmiranje u slučaju ozbiljnijeg poremećaja rada, a sistem menija olakšava konfigurisanje osnovnih parametara. Olakšice za upravljanje mrežom su dostupne preko *master* i *slave* priključaka. Cena uređaja je 495 britanskih funti.

Multimedija

Pravi 3D zvuk za kancelariju



Ukoliko Vam ne treba nova zvučna kartica ali niste zadovoljni zvukom možda treba da potražite bolje zvučnike Jedno od kvalitetnijih rešenja je novi proizvodameričke firme *NuReality* **Vivid3D Speaker** sistem **Vivid3D** zvučnici imaju ugrađena pojačala daju kvalitetan 3D zvuk i mogu se koristiti sa računarskim zvučnim karticama portabl radio ili kasetnim uređajima CD plejerima kućnim stereo sistemima, mono ili stereo televizorima ili sistemima za video igre Dimenzije zvučnika su 9 4 * 4 8 * 8 6 inča (V*Š*D) i teže po 2 kilograma svaki a pojačalo je snage 10 W po kanalu Svaki zvučnik poseduje po četiri magnetna zvučna drajvera i priključuje se za izvor zvuka pomoću priloženih kablova i stereo mini priključka. Kontrole na zvučniku omogućavaju regulisanje snage/jačine ulaznog zvora, centriranje (koje se fokusira na primarne zvučne elemente) i prostornu komponentu (ukupnu zvučnu daljinu) a kada se uključi SRS pojačanje razlika u punoći i realnosti zvuka dolazi do punog izražaja Povećani prostorni kvalitet zvučnika je dovoljan da ne biste poželeli da ih se odreknete i vratite na posao posle ovih modela svakako više nećete moći da se vratite na jeftine tajvanske zvučnike

Vivid3D zvučnici se isporučuju u crnoj ili standardnoj računarskoj sivoj boji a cena jednog para je 199 95 američkih dolara

SOFTVER

Worl Wide Web

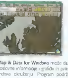
Shareware HTML editor za Macintosh



PINK Software je napravio **PINK Web Manager** editor koji pravi HTML fajlove za Web Page **Web Manager** je napravljen za Apple *Macintosh* i očitava ASCII tekst i PIC slike Naravno program omogucava *Hyperlink*-ove sa drugim dokumentima Softver se distribuira po *shareware* koncepciji

Poslovni programi

Prikazivanje podataka na geografskoj mapi



Map & Data for Windows može da locira poslovne informacije i grafički ih prikaže u *Windows* okruženju Program podržava uvoz baza podataka ili tabela iz standardnih programa kao što su *dBase Access Paradox, Excel* i *Lotus 1 2 3* Program dozvoljava prikaz podataka za različite oblasti na mapi u različitim bojama simbolima ili graficima u obliku "pite ili histograma a moguće je formirati i upravljati sa do 64 sloja na mapi Program se isporučuje sa nizom mapa koje sadrže linije obale administrativne granice vodotok mrežu puteva bitnije gradove i aerodrome Postoje dve jezičke varijante engleska i na francuska Proizvođač programa je francuska firma *A D D E.*, a cena softvera je 495 britanskih funti

Edukativni softver

Naučite kineski ... pomocu PC-ja!



Kineski proizvod zasnovan na jeftinoj radnoj snazi i savremenim tehnologijama se sve više probijaju na svetskom tržištu a kineska ekonomija je u ekspanziji koja odudara od recesije koja je planetarni proces u razvijenim zapadnim i postkomunističkim istočno evropskim zemljama Zbog nezadrživog porasta standarda vrlo intenzivnog uvođenja elektronike u proizvodnju Kina se računa kao jedno od tržišta sa trenutno najvećom ekspanzijom Kineski jezik i na dalekom istoku i na zapadu postaje sve ... izučavanje pa je američka ... *Inc* nedavno pustila u prodaju ... danje udžbenika za ovaj jezik.

Program **Professional Interactive Chinese for Windows version 2 0** je prvi interaktivni paket za učenje mandarinske varijante kineskog jezika Ono što nedostaje programu je pregledni vizuelni izgled ali on biva nadoknađen snagom i organizacijom sadržaja. Program je prevashodno namenjen početnicima i srednjim poznavaocima kineskog jezika Kao bitnije sadržaje CD ROM-a vredi pomenuti rečnik sa mogućnošću dodavanja novih reči vežbanje pisanja znakova fonetske kartice i vežbe i igre koje služe kao podrška lekcijama. Animacije i prepoznavanje govora pomažu polazniku da usavrši pravilan izgovor kineskih glasova Cena programa **Professional Interactive Chinese for Windows version 2 0** je 295 američkih dolara

Antivirusni programi

Lov na viruse pod Windows NT

Sophos Sweep for Windows je program koji služi za lov na viruse u *Windows* mrežnom okruženju a odnedavno se pojavila i verzija koja podržava rad sa *Windows NT* mrežama Program dozvoljava upravniku mreže da direktno ispituje prisustvo *DOS Windows* virusa na serverima i radnim stanicama širom mreže *Intercheck* modul programa **Sweep** omogućava proveru virusa za svaki PC koji je priključen u mrežu preko verifikacije svakog nepoznatog programa pre nego što mu dozvoli da se izvrši Program je proizvela britanska firma *Sophos Place* a cena ovog proizvoda je 495 britanskih funti godišnje za do 25 korisnika ili 895 funti za više od 25 korisnika

Virusi

Makro virus koji napada Winword 6 0

Korisnici računara se svakodnevno susreću sa raznim virusima neki napadaju *boot sector* neki se linkuju na izvršne fajlove a neki se šire putem makroa Dobro ste pročitali! Novi revolucionarni virus je preznojio mnoge korisnike najrasprostranjenijeg tekst procesora za *Windows* operativni sistem *Word for Windows* 6.0 Virus je u potpunosti bezopasan i izuzetno lak za saniranje ali je ipak zaprepastio korisnike personalnih računara naime niko nije očekivao ovakav način širenja virusa Iz *Microsoft*-a javljaju da virus makro inficira svaki *Word* 6.0 dokument kačeći se za NORMAL DOT podrazumevani *template* koji je otvoren uz većinu novih *Word* dokumena

a virus upozorava šta radi i kakvu štetu je mogao da napravi

Postoje nepotvrđene izjave da je kompanija čak nepažnjom distribuirala neke CD-ROM ove sa virusom. *S & S International* je napravio program koji isključuje automatsko izvršavanje makroa pod programom *Word for Windows* 6.0. Program može da se nađe na *S & S Web* site na **http://www/sands.com**. Do sada su prijavljeni slučajevi pojavljivanja virusa širom sveta: u Velikoj Britaniji, SAD, Rusiji, Finskoj, Nemačkoj, Holandiji, Francuskoj, Kanadi i Turskoj. Bez obzira što virus nije destruktivan, izgleda da je izazvao mnogo glavobolje onima čije je računare zarazio.

Uslužni softver

Bolje upravljanje bibliotekama fontova

Grčka firma *Magenta Ltd.* napravila je program **Font Director** koji pomaže dizajnerima i grafičkim umetnicima da instaliraju vlastite fontove grupisane po nameni, čime se olakšava upravljanje bibliotekama fontova. Pomoću ovog programa je moguće učitati i izbaciti fontove, a zatim obrisati bilo koje FOT fajlove koji se ne koriste, u cilju optimizacije korišćenja prostora na hard disku. **Font Director** vrši i prikaz izgleda znakova fonta i olakšava korišćenje karaktera kojim se ne može pristupiti pomoću tastature. Takođe je omogućeno i štampanje uzoraka i tabela rasporeda karaktera na printeru. Cena programa je 70 američkih dolara.

Komunikacije

Razmena binarnih fajlova faksom



Pomoću programa **3D Fax** moguće je slati binarne fajlove za editovanje ili izvršavanje preko standardnog faksa. **3D Fax** podržava kompresiju i procesiranje slika, što omogućava da se redukuje vreme slanja fajlova za najmanje 90 procenata, a moguće je slati i kolor slike i multimedijalne fajlove. Kada se za prijem koristi faks mašina, otisak koji se dobije pomoću uređaja je vizuelno neupotrebljiv, ali se nakon skeniranja, po tvrdnji proizvođača, američke firme *Infolmaging Technologies*, pomoću **3D Fax** softvera skenirani fajl restaurira u prvobitan sadržaj fajla koji je poslat. Nakon toga je je fajl spreman za pregled i obradu programima za koje je namenjen. Standardna verzija programa **3D Fax** košta 99 američkih dolara, a profesionalna verzija **3D Fax Pro** se dobija za 199 dolara.

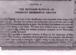
Naravno, ostaje skepsa nad kvalitetom dobijenih fajlova koji se prenose između ostalog i štampanjem na termalnom papiru i njegovim skeniranjem. Bitno je da je neko pokušao da automatizuje i ovakvu vratolomiju, a koliko su rezultati upotrebljivi je sasvim drugo pitanje.

Obrada teksta

Konačno TeX za Windows

Opšte prihvaćeni standard za formatiranje i pisanje kompleksnih naučnih radova na zapadu je svakako TeX. Dosta dugo korisnici nisu mogli komforno da rade u TeX okruženju pod operativnim sistemom *Windows*. Konačno su se na američkom tržištu pojavili programi **PCTeX for Windows** firme *Personal TeX Inc.* i **TrueTeX for Windows** firme *Kinch Computer Co.* Ovi programi predstavljaju neophodnu alatku za pripremu velikih dokumenata pretovarenih jednačinama, tabelama i graficima.

Oba programa objedinjavaju tri osnovne funkcije svakog TeX programa: kompajliraju TeX binarne DVI fajlove, služe za pregled sadržaja DVI fajlova i njihovo štampanje. Međutim, ovi programi se znatno razlikuju se po pitanju integracije i korisničkog *interface*-a. Oba konkurenta sadrže desetine font fajlova za *Computer Modern Roman* znakove u širokoj skali različitih veličina. **TrueTeX** nudi 32-bitno kompajliranje, pregledanje i štampanje u *Windows* okruženju. **PCTeX** iz *Windows* okruženja poziva DOS *compiler*, a pregled i štampanje vrši direktno iz *Windows*-a. I jedan i drugi program podržavaju tri popularne varijante TeX-a: LaTeX, AMSTeX i AMSLaTeX.



PCTeX poseduje bolji korisnički interface: iz programa je moguće zvati compiler, viewer i program za štampanje. Dijalozi su tako dizajnirani da ako projekat zahteva DVI fajl, onda se on pojavljuje kao podrazumevan kada se pozove Print DVI ili View DVI potprogram iz **File** menija. Sve opcije ovog programa su dostupne preko dobro dizajniranog menija sa alatima. Međutim, **PCTeX** se nezgrapno oslanja na DOS boksove za pozivanje compiler-a. Compiler se isporučuje u dve verzije: prva je za standardni rad i druga za rad sa DOS extender-om.

Za razliku od prethodnog programa, **TrueTeX** pruža integrisani *Windows* pristup radu, ali sa posebnim programima za kompajliranje, pregled i štampanje TeX dokumenata. Sve u svemu, korisnički interface nije elegantan kao kod konkurenta. Na primer, ukoliko se prebacujete iz običnog TeX-a u LaTeX, program zaboravlja direktorijum u kojem čuvate TeX fajlove. Prednost ovog programa je što njegov viewer može da pretražuje sadržaj DVI fajlova.

PCTeX se pri kompajliranju na 486DX2/66 računaru pokazao kao nešto malo brži, a pravo pitanje na osnovu kojeg se odlučuje između dva programa je da li korisnik preferira integrisani pristup, gde nesumnjivu prednost ima **TrueTeX**, ili bolji interface, kada pobeđuje **PCTeX**.

Program **PCTeX for Windows 2.0** firme *Personal TeX Inc.* košta 299 američkih dolara, a **TrueTeX for Windows 4.0** firme *Kinch Computer Co.* 450 dolara.

DOMACE VESTI

Skupovi

Zaštita podataka u računarskim sistemima

Savez inženjera i tehničara Srbije organizovao je 16. i 17. novembra 1995. u prostorijama Centra "Sava" stručni skup pod nazivom "Zaštita podataka u računarskim sistemima". Cilj skupa je bio da se sagleda nivo zaštite podataka u našim uslovima, obuhvativši pri tome sve globalne aspekte zaštite: sistemsku, fizičku, organizacionu i pravnu. Na skupu je održano trinaest izlaganja, koja su pokrila različite teme: sigurnost UNIX sistema, logičko-tehničku zaštitu, računarske viruse, kriptologiju, sigurnost i zaštitu podataka kod elektronskih novčanih transakcija, zaštitu od računarskog kriminala, pravne aspekte zaštite podataka kao i zaštitu ličnih podataka. Skup je okupio širok dijapazon zainteresovanih, od stručnjaka koji se neposredno bave ovom oblašću pa do onih koji su skup iskoristili za njeno upoznavanje. Organizator je okupio brojnu ekipu sponzora, među kojima su bili *SAGA, SDC, CIP, AB soft, Digit, Vidra info, BiznisLink* i *Informatika*. ■

Dostignite visine



sa nama ....

BIT computers - bitno drugačiji

486DX4-120

memorija 4 MB
hard disk 850 MB
flopy disk 1 44 MB
kontroler VLB EIDE
kolor monitor 14" L R
vga 1MB Trident 9440 VLB
mini tower case,
tastatura CHICONY YU
miš + podloga
instalacija
* MS-DOS 6 2
* Win 3 11
* MS OFFIC

1,795.-

PENTIUM 100

memorija 32 MB
hard disk 1.1 GB,
flopy disk 1 44 MB,
kontroler VLB EIDE
kolor monitor 17" PHILIPS,
vga 2MB Stealth 64 VLB
mini tower case,
tastatura CHICONY YU
miš + podloga,
instalacija
* MS DOS 6 2
* Win 3 11
* MS-OFFIC

4,990.-

doplate:	
sa 850 na 1 1 GB	60
flopy disk 1 2 MB	65
RAM sa 4 na 8 MB	195
RAM sa 4 na 16 MB	585
UPS 500 VA	430
UPS 1000 VA	730
Epson LX300	370
Epson Stylus 800+	630
Epson Stylus color	1220
CD ROM double speed	185
Sound blaster 16 ASP	210

bit computers

narodnog fronta 53, novi sad, tel/fax 021 369 191, tel 365 222 lok 179, 175
cene se menjaju - proverite ih na dan odluke o kupovini


sledeća generacija:
potpuno oslobođenje...
STIL computers
Njegoševa 42, Beograd,
tel/fax 444-15-65, 451-982
prepress Studio 6&2 • produkcija UltraArt

CompFair 95

Duh zapada uveliko se oseća u Mađarskoj. Sajam računara CompFair je već postao poznat izvan granica našeg severnog suseda, privlačeći veliki broj poslovnih ljudi i računarskih zanesenjaka koji su i ove godine krenuli put Budimpešte.

Goran Alimpić

Dok je autobus prolazio predgrađima ulazeći u gradsku vrevu, zapanjeno sam gledao kako promiču šarene reklame, moderni automobili i blještave prodavnice sa vitrinama prepunim luksuzne robe. Pre samo nekoliko godina sve je izgledalo sasvim drugačije. Sloboda i zamah privatnog kapitala preporodili su celu zemlju, tako da sam morao, uz dozu zavisti, da priznam da su sa nama zamenili mesta.

Veoma je vidljiv nagli rast upotrebe modernih tehnologija. Na svakom koraku očekivale su me automatske menjačnice novca i prodavnice računarske opreme u kojima se mogu lako naći sve komponente u originalnim pakovanjima i uz punu garanciju. Iako se i daje bon sa softverskim piratstvom, Mađarska je otvorena za strana predstavništva i legalne dilere. Lokalnih računarskih časopisa ima bezbroj, a u većini trafika mogu se naći i originalna izdanja *PC magazine*-a, *Byte*-a i ostalih najvećih izdavača ove literature.

U istočnom delu grada smešten je sajamski kompleks *Budapesti Nemzetközi Vásárközpont*. U njemu se održavaju sve najveće manifestacije, uključujući i tradicionalni *CompFair* sajam računara, opreme, programa. Uporedo sa njim, u nekoliko manjih hala paralelno su bile na programu druge zanimljive priredbe, kao što je sajam marketinga.

Iako je *CompFair* sigurno najveća manifestacija ovog tipa u Istočnoj Evropi, ono što je odmah vidljivo je da joj nedostaje glamur zapadnih sajmova na kojima lete konfete, dele se čokolade i sklapaju milionski ugovori. Ako zanemarimo količinu izlagača i gabaritne sale, ceo prostor na prvi pogled podseća na nama dobro znani beogradski Sajam tehnike.

Veliki i mali

Najveći i najuočljiviji prostor je zakupio pogađate, *Microsoft*. Ceo jedan blok pokriven je platnom sa oslikanim nebom, što uočljivo podseća na njihov udarni proizvod *Windows 95*. Svaki ugao ovog velikog štanda imao je posebne programe: na jednom mestu su se održavale prezentacije, na drugom su radoznali mogli da isprobaju novi softver, na trećem je bio šalter za informacije... Oko štanda je uvek vladala velika gužva. *Microsoft*ovi službenici bili su veoma uglađeni, lepo odeveni i perfektno su govorili engleski, za razliku od drugih mesta na kojima je znanje nekog svetskog jezika, na moje iznenađenje, bila prilično retka pogodnost.

Kada sam napokon uspeo da se izvučem iz gužve i krenuo u potragu ka sledećim ciljevima, zamalo sam promašio štand u neposrednoj blizini na kome se predstavljao OS/2. Nije ni čudo, bio je potpuno pust.



*Borland*ov štand je bio na drugom mestu po originalnosti: izrađen u antičkom stilu, neodoljivo je odavao svoju glavnu poruku: *Delphi*. Međutim, ljubazne domaćice nisu imale da ponude ništa novo - na pitanje o *Delphi*ju za *Windows 95*, samo su slegle ramenima.

Pažnju je privlačio i štand firme koja predstavlja veći broj značajnih kompanija, među kojima je i *Epson*. Tu se nalazio zanimljiv eksponat: providna verzija kolor štampača *Stylus*, popularnog u našim krajevima, koji je, uz mogućnost da se zaviri u unutrašnjost njegove konstrukcije, neprestano štampao propagandne plakate. U blizini se nalazio i *Panasonic* sa novom serijom svojih univerzalnih mašina koje u sebi sadrže štampač, telefon, sekretaricu, faks i modem.

Slobodno isprobavanje robe je bila jedna od lepih karakteristika celog sajma, preuzeta sa velikih izložbi ovog tipa. Na primer, na nekoliko mesta moglo se krstariti Internetom i sanjati o vremenima kada će to biti dostupno i iz našeg podneblja.

Jednu pravu novost na ovom Sajmu donela je poznata mađarska firma *Recognita*. Ona je ekskluzivno predstavila novu verziju svog poznatog programa za prepoznavanje štampanog teksta *Recognita Select 3.0*. Iako je punoletstvo ovog paketa donelo je mnoštvo kozmetičkih izmena, ali malo napretka u procentu uspešno pročitanih karaktera. I pored velikog truda gospodina koji je specijalno za vašeg izveštača izveo detaljnu prezentaciju, programu se nisu mnogo svidela slova iz mađarskog dnevnog lista.

Raj za kupce

Pored velikih, u halu se naguralo i mnoštvo malih lokalnih firmi. One su uglavnom nudile hardver i softver sa specijalnim sajamskim popustom. Stvari su se prodavale po principu „plati i nosi", a velike gužve oko prodajnih mesta su jasno stavljale do znanja da su Mađari znatno popravili svoju kupovnu moć. Cene hardvera su znatno niže nego kod nas, čak i do 50%, uz neuporedivo kvalitetniju uslugu i podršku.

Vraćajući se u Beograd, uz kesu punu prospekata koje su mi prosto gurali u ruke još od ulaska u BNV kompleks, znao sam da se vraćam tamo gde će sve ovo što sam video u Budimpešti biti san još neko vreme. Poželimo samo da to vreme bude što kraće. ■

Informatički podmladak

Na Zlatiboru se, početkom novembra, održao skup sa temom "Studentski radovi u informatici i računarskim naukama". Za četiri dana, koliko je SinFON trajao, pokazalo se da za budućnost domaće informatike ne treba da brinemo

Vladan Aleksić

[illegible] (K[illegible]zba) ukazuje na razlog zašto mno[illegible] od poslovne aktivnosti. Izlazak [illegible] sfere, promena lokacije i ve[illegible] u jednom me[illegible] [illegible]

[illegible] gram, nedostatak udžbenika i pre svega neopremljenost obrazovnih ustanova odgovarajućom opremom odbili su mnoge od računara kao usmerenja.

Na putu ka Zlatiboru ruku mi je dopao Zbornik apstrakata, tako da sam imao prilike da pre samog početka skupa saznam nešto o radovima koji će biti izloženi. Već posle pročitanog sadržaja primetio sam jasnu razliku od suvoparnih [illegible] izlaganja sa kojima sam očekivao da ću [illegible] mogu videti da svi [illegible] vrednost. Poen[illegible]

[illegible]

SinFON [illegible] Kao sr[illegible] se **Ministarstvo za nauku i tehnologiju Republike Srbije**, najbolji pokazatelj da [illegible] razvoj ove grane nauke bilo je prisustvo zamenika ministra, gospodina Miloška Lazi[illegible] ovaj simpozijum ne bi se mogao održati da nije bilo mnogobrojnih sponzora; navešću samo one koji su bili zapaženi svojim prezentacijama. **Ei Sigral**, zastupnik Silicon Graphics a za Jugoslaviju, predstavio je servere ove firme. **OTC computers** zadržao je pažnju slušalaca na komercijalnim WWW prezentacijama, a **JP PTT Srbije** je izložio neke aspekte planiranja razvoja IS u složenim poslovnim sistemima.

Prijavljeno je ukupno 169 radova, od kojih je posetiocima izloženo 148, podeljenih u nekoliko grupa. **Informacioni sistemi** su obuhvatili niz radova iz problematike projektovanja IS, dok se **Primena informacionih sistema** bazirala na konkretnim zadacima: sistemu tehničke zaštite, kontroli [illegible] vodovoda, praćenju referenci zapo[illegible] Objektno orijentisane baze i CASE [illegible] najveći deo izlaganja iz grupe **Baze podataka** [illegible] se **Arhitekturi računara** [illegible] optimizacijama proceso[illegible] kontroli. **Paralelno procesiranje** donelo je niz zanimljivih i [illegible] **Računarske mreže i protokoli** [illegible] posvećeni Unix-u, što je [illegible] da je to prirodno okruženje [illegible] akademske populacije. U **Programskim jezicima i programiranju** preovlađivala je računarska grafika (3D, CAD/CAM i multimedija), dok su se radovi u **Simulaciji i simulacionim jezicima** kretali u rasponu od simulacionih analiza pa sve do računarskih animacija. **Veštačkom inteligencijom i ekspertnim sistemima** dominirale su neuronske mreže, dok je u **Sistemima za podršku odlučivanju** prikazano nekoliko metoda u kojima su korišćeni alati od CASE-a do Excel-a. Grupa **CIM** se, kao što joj naziv govori, bazirala na automatizaciji proizvodnje, dok se **Statistika** bavila analizom tržišta, korespodencije populacije i sl. I na kraju, **Operaciona istraživanja** su okupila niz zanimljivih radova iz oblasti komparativne analize, mrežnog planiranja, linearnog programiranja i dr.

Moderatori grupa su predložili najistaknutije radove Organizacionom odboru, a časopis "Računari" se uključio u sponzorisanje najboljih poklonivši svakome po jednu godišnju pretplatu.



* * *

Prisustvujući predavanjima, stekao sam mnogo utisaka. Kao prvo, primeti se da su predavači neiskusni u prenošenju svog znanja na publiku, nedostaje im prodavačko umeće starih informatičara, izbrušeno u domaćim firmama ili praksi svojih profesora. Međutim, sva to zamuckivanje i nesigurnost nestaju pred jednom činjenicom, a to je da iza tih mladih stoji znanje i veliki potencijal. Sa sankcijama ili bez, informatika ove države je u dobrim rukama.

Pitanje je samo da li je ona toga svesna. ■

Domaći specijaliteti

Na sajmovima i prezentacijama kod nas obično nismo u prilici da vidimo dostignuća domaćih firmi u oblasti računarske tehnike i informatike. Umesto toga, bivamo zaglušeni velikom količinom buke i mnoštvom različitih ponuda istih stvari. Izuzeci ipak postoje.

Nenad Mitovski

U hotelu *"Beograd Intercontinental"* je od utorka 24. do petka 27. oktobra ove godine održan treći po redu *"Microsys Computer Show"*. Ulaz je bio slobodan, a posetioci su mogli da prisustvuju zanimljivim demonstracijama, prezentacijama, da čuju stručna predavanja, pa čak i da učestvuju na okruglim stolovima. Kao motiv za organizovanje ovakve manifestacije ljudi iz *Microsys*-a su istakli činjenicu da kod nas ne postoji odgovarajući ambijent za ozbiljnu prezentaciju projekata i rešenja u oblasti informatike. Bez obzira na opštu atmosferu izgovaranja sankcijama, ekonomskom situacijom, kao i na to da je za balkanskog čoveka uobičajeno da teško pohvata praktična i korisna rešenja, ova firma se nije preterano bavila pitanjem da li računare treba koristiti, već je pokazala kako ih treba koristiti i primeniti u poslu.

Kao što je i red, ceo skup je započeo u svečanoj atmosferi, uz prigodan koktel, a onda...

Vežite se

Radni deo počeo je predavanjem gospodina Kalmana Cinklera iz PTT Srbije. Cinkler je govorio o informacionoj mreži PTT-a, kao i o mogućnostima pristupanja ovoj mreži i informacijama pomoću najobičnijeg PC računara. Akcenat je stavljen i na funkcionisanje heterogene mreže računara, a s tim u vezi je, nakon predavanja, napravljena i demonstracija rada jedne takve mreže postavljene od strane *Microsys*-ovih stručnjaka. U mreži su se nalazili računari različitog tipa, pod različitim operativnim sistemima (*MicroVAX* pod *VMS*-om, *Novell* i *Unix* serveri, *MS DOS/Windows* radne stanice...), povezani na razne načine, od obične modemske veze, preko *Ethernet*-a, do optičkih kablova. Šema mreže je sve vreme bila izložena na jednom povećem panou. Povezivanje i umrežavanje računara istaknuto je kao neophodan preduslov da bi se ulaganje u računare zaista isplatilo. Izolovani računar je vrlo često samo skupa kancelarijska mašina, dok mreža računara predstavlja temelj za postavljanje distribuiranih baza podataka, složenih informacionih sistema i svega onoga što je prezentirano u narednim danima.

A karton!?

U sredu je predstavljen *YUBIS*, kompletan bolnički informacioni sistem razvijen za potrebe KBC "Bežanijska kosa". Ovaj projekat su zajedno radile firme *"MS Microsys"* i *"CIT inženjering"*, a jedna od njegovih bitnih karakteristika je ta da se lako mogu generisati modeli informacionih sistema i za sve ostale tipove zdravstvenih ustanova. U skladu s ovim, podnevno predavanje imalo je za temu efekte primene informacionih sistema u medicini, a držao ga je načelnik sektora za informatiku VMA Milovan Obradović. Nakon toga organizovan je okrugli sto na kojem su zainteresovani posetioci, pretežno zdravstveni radnici, diskutovali o automati[illegible] bolesti, opštem problemu [illegible] zdravstvenih ustanova, kao i [illegible] ponuđenim rešenjima.

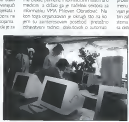


Foto: Goranko Matić

48 sati u...

Treći dan je bio posvećen platnom prometu, a u skladu s tim u 11 sati je demonstriran rad programskog paketa za bankarsko poslovanje *BIS*. Akcenat je stavljen na modul za platni promet koji je razvijan na bazi koncepcije ponuđene od strane ekspertske grupe guvernera NBJ dr Avramovića. Ekspertska grupa radi u sklopu aktivnosti za rekonstrukciju monetarnog sistema. U podne je predavanje na temu "Savremeni tržišni platni promet" održao profesor ekonomskog fakulteta u Beogradu Života Ristić, a efikasna primena računara je istaknuta kao neophodan uslov za uključenje u svetske finansijske tokove.

Za okrugli sto na temu "Platni promet u poslovnom bankarstvu" verovalo se da će biti nešto burniji od onog o zdravstvu - očekivana je živa diskusija između "bankara" i "SPP-ovaca". Bankari su diskutovali o tehničko tehnološkim i organizacionim aspektima prelaska platnog prometa u ruke banaka, ali očekivane reakcije iz SPP-a nije bilo.

Progres

Poslednji dan bio je rezervisan za *IPS* (*Integralni Poslovni Sistem*), projekat koji je predstavljen prošle godine na istom mestu. U međuvremenu je ovaj paket našao svoje korisnike, razvijan je i dorađivan u skladu sa njihovim različitim zahtevima. Demonstriran je i rad ovog sistema u *client-server* mreži heterogenog tipa sa distribuiranom bazom podataka, a održan je i godišnji skup korisnika *IPS*-a.

Mogli smo čuti i dva predavanja. Prvo, "Koncept savremenog poslovnog informacionog sistema" držao je Slobodan Ivošević, a drugo, na temu "Razvoj strateških aplikacija - *Progress* ver. 7" održao je Staniša Stamenković iz firme *"ABV informacioni sistemi"*. Ovakva tematika je svakako privukla i ljude iz računarske struke, programere posvećene razvoju relacionih baza podataka. Odluka firme organizatora cele ove manifestacije da za razvoj svojih aplikacija koristi *Progress* potkrepljena je vrlo jakim argumentima, kao što su odnos performanse/cena (gde se pod cenom podrazumevaju i hardverski zahtevi), prenosivost na skoro sve postojeće operativne sisteme, kao i mnogi drugi.

• • •

Microsys Computer Show "95" posetilo je oko 1000 registrovanih gostiju različitog profila, od ljudi kojima su računari profesija do onih koji tek počinju da ih primenjuju u svojoj struci - zdravstvu, bankarstvu, privredi. Zalutalih nije bilo. Profil posetilaca odgovarao je temi rezervisanoj za taj dan, a sve vreme bilo je moguće i sesti za neki od računara u mreži i probati kako sve to radi. Sve je bilo na odgovarajućem nivou i vrlo ozbiljno, čemu je svakako doprinela i specifična atmosfera hotela "Beograd Intercontinental". ■

Autor je urednik emisije o računarima "PC PLUS" na Radio Indexu. Možete ga kontaktirati na e-mail adresu oken@sezam.co.yu

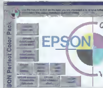
Boja na stolu

Jeftin kolor štampač, sa visokim kvalitetom reprodukcije fotografija je skora budućnost *Epson*-ovi modeli *Stylus Color Pro* i *Stylus Color Pro XL* prvi su odškrinuli vrata u tu budućnost

Milan Četić

Retki su oni korisnici računara koji bar jednom nisu poželeli da na svom stolu imaju mogućnost štampe u kvalitetu koji se ne razlikuje od onoga koji se postiže u štamparijama. Od pre desetak godina, kada su se pojavili prvi komercijalni štampači u boji, do danas tehnologija štampanja i štampača polako se približavaju tom cilju. U procesu iznalaženja rešenja bilo je mnogo slepih ulica, neupotrebljivih tehnologija i nezgrapnih proizvoda. Do danas su se jasno iskristalisali pravci kojima će se kretati razvoj tehnologije, tako da se uz malo pažljivije biranje, može odabrati sasvim pristojan štampač koji će biti veoma upotrebljiv barem dve-tri godine.

U principu postoje tri dominantne tehnologije koje omogućavaju štampanje u boji, od kojih je svaka jasno namenjena određenom segmentu tržišta. *Dye sublimation* tehnologija daje najbolji kvalitet reprodukcije fotografija koji trenutno postoji, ali je zbog visoke cene otiska i samih štampača u upotrebi skoro isključivo u visokoprofesionalnim mašinama u grafičkim biroima i štamparijama za izradu probnih otisaka pred štampu. *Laserska* tehnologija na sadašnjem nivou razvoja, zbog veoma niske cene eksploatacije i velike brzine rada, ima namenu prvenstveno u slučajevima kada je potrebno odštampati veliku količinu materijala u boji, pod uslovom da kvalitet reprodukcije fotografija nije preterano važan. *Ink jet* tehnologija je namenjena prvenstveno za ličnu i *SOHO* (*Small Office - Home Office*) upotrebu. Odlikuju je niska cena štampača i razumna cena potrošnog materijala. Kvalitet reprodukcije fotografija varira, u zavisnosti od modela, od lošeg do zadovoljavajućeg. Ostale tehnologije (*wax transfer, solid ink*...) su veoma malo rasprostranjene i praktično su pred izumiranjem.



Izgled uvodnog ekrana instalacije EPSON Perfect Color Pack II

Koncepcija

Samo ime firme *Epson* asocira na najrasprostranjeniji standard na tržištu matričnih štampača. Osim matričnih, *Epson* proizvodi i laserske štampače, a pre oko dve godine počeo je da proizvodi i *ink jet* štampače. Za razliku od ostalih proizvođača koji koriste *termalnu*, *Epson* je doneo odluku da koristi *piezo* tehnologiju glave za štampanje.

Osnovna razlika između ove dve tehnologije je u tome što se kod termo glave mastilo za štampanje izbacuje zagrevanjem, dok se kod piezo glave koristi piezoelektrični efekat. Njene prednosti su u tome što se mastilo ne greje prilikom štampanja (i zato ne mora da bude otporno na visoke temperature) i što je lakše vršiti doziranje mastila prilikom štampanja. Loše osobine su nešto manja brzina štampanja i visoka cena glave za štampanje. Zbog toga se piezo glava ne menja zajedno sa kertridžima sa bojom, već je poseban deo štampača i mora se menjati posle određenog broja odštampanih znakova, što dodatno povećava troškove održavanja.

Međutim, opredeljenje za ovakvu koncepciju samo pokazuje da su u *Epson*-u odlučili da glavna osobina *Stylus Color Pro* i *Stylus Color Pro XL* štampača bude kvalitet. Ovom cilju su podređeni svi ostali parametri konstrukcije, i to se vidi već na prvi pogled.

Opšti utisak

Kao prvo, nužno se nameće poređenje sa starijim modelom *Stylus Color*. Spoljni izgled je malo poboljšan, tako da *Color Pro* i *Color Pro XL* izgledaju robusnije nego njihov prethodnik. Umesto stare police koja se izvlači dodato je nešto što liči na pravu fioku za papir sa poklopcem i graničnicima. Stari način ulaganja papira, pri kojem dolazi do previjanja od 180 stepeni, nije promenjen. Zbog ovoga nije moguće štampati na papirima težine veće od 120 g/m², jer dolazi do veoma izraženog *banding*-a (ovo je pojava koja se javlja prilikom štampanja kada se zbog neispravnog pomeranja papira razdvajaju prolazi glave za štampanje, pa odštampana slika izgleda kao da je sastavljena iz traka).



Što se tiče odluke da se istovremeno na tržištu pojave i A4 (*Color Pro*) i A3 (*Color Pro XL*) model, smatram da je to veoma dobar potez. Kao prvo, sama pojava A3 modela je veoma značajna jer je to tek treći (pored *HP*-a i *Canon*-a) A3 *ink-jet* kolor štampač na tržištu. Drugo, relativno niska cena oba modela omogućava da budu pristupačni i onima koji do sada nisu mogli ni da sanjaju kvalitetnu kolor štampu. Model A3 može da štampa formate veće od A3 (do 328x486 mm), što ga uz male margine (4mm sa svih strana osim 13 mm sa donje strane) čini jednim od retkih štampača u boji uopšte (uključujući tu i laserske i *dye sublimation* štampače) koji

mogu da odštampaju fotografije veće od punog A3 formata

Međutim i pored malo ojačane konstrukcije *Color Pro* štampači su i dalje amaterske (lične) ili u najboljem slučaju poluprofesionalne mašine. Mehanizam za povlačenje papira "kolica" na kojima se nalazi glava za štampanje i sama glava za štampanje svojim kvalitetom nikako ne zadovoljavaju potrebe koje pred njih postavlja profesionalna upotreba. Takođe mala brzina i mali predviđeni obim štampanja govore da se radi o štampaču malih pretenzija. Ako bismo pravili poređenje sa automobilima, ond bi se za ova dva modela moglo reći da su sportski kupei zgodni za pokazivanje i kratke ture ali praktično neupotrebljivi za stalne i duge vožnje

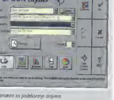
Štampanje

Ono u čemu su ova dva štampača najači je reprodukcija fotografija. Ne računajući najfinije tonske prelaze u svetim delovima slike teško je primetiti razliku između fotografije odštampane na Epson u i one odštampane u štampariji. Ovo zvuči neobično s obzirom na to da je rezolucija štampanja samo 720 tačaka po inču. Ali kada znamo da *Color Pro* štampači koriste FM (frekventno modulisanu) rasterizaciju stvari postaju mnogo jasnije (pogledaj tekst "Kako se formira slika"). Osim štampanja u 720 tpi *Color Pro* nudi mogućnost štampanja u normalnom (360 tpi) i ekonomičnom (180 tpi) režimu. Takođe štampa je različita kada radite na običnom (fotokopir) papiru ili na dve vrste specijalnog (za rezolucije od 720 i 360 tpi) papira. U stvari kad kombinujemo razne rezolucije i razne vrste papira dobijamo veliki broj režima štampanja što nas dovodi do drajvera



Drajveri su stara *Epson* ova boljka koja je bila posebno izražena u *Stylus* seriji. Ali ljudi iz *Epson* a su konačno ozbiljno shvatili da je to veliki problem za prodaju njihovih štampača i rešili su da za jednom za svagda reše. Zato CD ROM pod nazivom "*EPSON Perfect Color Pack II*" koji sadrži kompletan softver koji dolazi uz *Stylus Color* seriju delu je impozantno

Na ovom CD ROM u dolazi kompletna podrška za sve *Stylus Color* štampače (*Color Color Pro* i *Color Pro XL*) u trojezičnoj varijanti (engleski francuski i nemački) kao i za crno belu *Stylus* seriju (modeli 400 800 800+ i 1000). Osim samih drajvera na CD u se nalaze sledeći programi i podaci



Izgled prozora za podešavanje drajvera

- Print Accelerator *program za zaobilaženje Print Manager a u Windows u radi bržeg štampanja*
- Color Copy *program koji ove štampače u kombinaciji sa EPSON skenerima pretvara u kolor kopir mašinu*
- Twain driver - *drajver za EPSON skenere*
- Print utilities *Razni programi koji uključuju i DOS drajvere kao i program za kalibraciju štampača*
- Tiff image files *datoteke sa demo fotografijama u TIFF formatu*
- Printer demos *pripremljene print datoteke sa demo fotografijama*

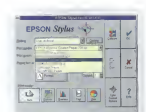
Pri pisanju drajvera vođeno je računa i o najsitnijim detaljima kao što je tip papira koji se koristi. Osim *Epson* ovog originalnog papira, predviđena je mogućnost štampanja na još dve vrste papira, oba od nemačkih proizvođača koji zadovoljavaju *Epson* ove kriterijume kvaliteta. Naravno ovo podešavanje nije samo deklarativno za svaki tip papira drajver podešava parametre štampe (kontrast i osvetljenje) što uzrokuje promenu količine odje koja se troši prilikom štampanja. Ono što se ovim postiže je maksimalno prilagođenje štampe fizičkim osobinama papira, a u cilju standarnog kvaliteta štampe

Uz korišćenje "*microweave*" tehnologije izuzetno je smanjen *banding* prilikom štampanja. Međutim i pored toga ako veoma pažljivo gledate sa malog rastojanja mali *banding* je ipak primatan što u suštini ne bi trebalo da predstavlja veliki problem

Naravno ova tehnologija dodatno usporava štampanje. Uopšte brzina rada nije osobina kojom se *Color Pro* može podičiti. Brzina štampanja A4 fotografije kreće se od 2 minuta u rezoluciji 360 tpi na običnom papiru do 15 minuta u rezoluciji 720 tpi na specijalnom papiru. Ova vremena su samo vremena štampanja i u njih nije uračunato vreme koje potroši računar za izradu rasterske slike. To vreme na računaru sa 486 procesorom na 100 MHz zna da bude i do četiri puta duže od samog vremena štampanja tako da za izuzetno kvalitetne fotografije može da protekne i po ceo sat od trenutka kada ste pokrenuli štampanje do trenutka kada iz štampača izađe odštampan papir

Što se DOS drajvera tiče ističe se drajver za *AutoCAD* mada postoje i drajveri za *Lotus Harvard Graphics Word Star* i *Word Perfect*. Drajver za *AutoCAD* urađen je odlično i konačno sadrži sve što je potrebno. Ovo podrazumeva mogućnost da se svakom od 255 pera dodeli zasebna odja i debljina. Debljine linija se za razliku od prethodnih verzija zadaju u milimetrima što je konačno omogućilo plotovanje u saglasnosti sa standardima. Male margine (po 3 milimetra sa strana, 7 sa gornje i 14 sa donje strane) omogućavaju plotovanje punog unutrašnjeg okvira A3 crteža na papiru A3 formata. Ovo je još jedna od osobina koje će mnogi korisnici kojima je dosadilo "petljanje" sa *oversize* papirima željno dočekati. Brzina štampanja je nešto veća nego prilikom štampanja fotografija. Za A3 format pri rezoluciji od 360 tpi potrebno je od 3 5 do 7 minuta, u zavisnosti od toga da li se štampa crno-belo ili u boji

Osim CD ROM a, uz štampač se dobijaju i dve kolor kartice za kalibraciju koje naravno imaju pravu upotrebu tek u kombina



[illegible] papira koje drajver prepoznaje

prosečnog CD-ROM uređaja je znatno manja od cene ovog štampača tako da ako ste već odlučili da kupite ovaj *Color Pro* ili *Color Pro XL* dodatni trošak za kupovinu neće mnogo uvećati iznos vaše investicije

Sve u svemu ...

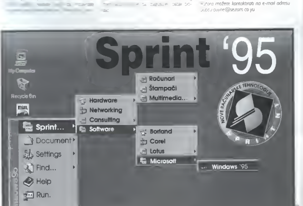
Ukupan utisak o ovim štampačima je povoljan Međutim njihova upotrebljivost je diskutabilna Kada je reč o malom arhitektonskom projektnom birou samostalnom arhitekti nezavisnom [illegible]

Ova serija štampača predstavlja dobro izbalansiran kompromis između cene brzine i kvaliteta Međutim taj kompromis je prilično ograničio ciljnu grupu korisnika kojima je ovakav štampač potreban Uprkos tome predviđam da će se ovi štampači bar izvesno vreme dobro prodavati ako ni zbog čega drugog a onda zbog toga što je prvi štampač koji je danas u stanju da pruži bolji kvalitet štampe fotografija oko deset puta skuplji od *Color Pro* i oko šest puta skuplji od *Color Pro XL* ■

Tehničke karakteristike (Color Pro XL):

Širina	580 mm
Dubina	597 mm
Visina	182 mm
Težina	10 kg
Instalisana snaga	20W
Vek kasete sa bojom kova (670 A4 strana pri 5% popunjenosti)	1 2 milijarde zna
Težina papira	od 64 do 90 g/m²
	0 08 do 0 11 mm

Autora možete kontaktirati na e-mail adresu [illegible]@sezam co yu



Kolor štampači - - tehnologije štampe

Vladimir Stakić

Pravljenje grafičkog izlaza pomoću štampača, u najvećem broju slučajeva, ima za cilj da informiše ili propagira. Dodavanjem kolora grafici otvaraju se nove mogućnosti, ali i dileme oko izbora najpodesnijeg modela štampača. Upoznavanje sa osobinama osnovnih tipova tehnologija koje se koriste u kolor štampačima, kao i mogućnostima njihove primene, sigurno će olakšati odluku.

U poslednje vreme kolor izlaz postaje neophodan u sve većem broju aplikacija, kao što su poslovne prezentacije, stono izdavaštvo (*DTP*), proba pred štampu, naučno tehničke prezentacije, *CAD CAM* dokumentacija, itd.

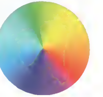
Svaki kolor štampač u sebi krije malu štampariju u boji. Svaka boja odvojeno se procesira i nanosi, a sve između konačnog rešenja na kompjuteru i otiska na štampaču procesira se elektronski. Iz ovoga se vidi i namena kolor štampača - jednostavno i brzo dobijanje kvalitetnog otiska.

Tehnologija štampanja, mehanizam koji se koristi za nanošenje boja na papir, određuje, više nego bilo koji drugi faktor, koje sve vrste poslova taj štampač može da obavlja. Članak razmatra pet komercijalnih tehnologija koje se najviše koriste u današnje vreme, i to: *inkjet*, *solid ink*, *thermal-wax*, *dye sub* i *laser*. Ovo su osnovni nazivi tehnologija. Međutim, često se koriste ili drugi nazivi ili se dodaju egzotična imena već postojećim tehnologijama, što ima čisto propagandni karakter.

Tržište kolor štampača nudi nekoliko konkurentnih tehnologija, kao i veliki broj različitih modela unutar njih. Radi jasnijeg sagledavanja prednosti i nedostataka tehnologija i njihovih modela uvode se sledeći pokazatelji:

- Okvirna cena štampača na domaćem tržištu. Uveden je radi čuvene izreke: "Koliko para, toliko i muzike".
- Cena materijala potrošenog pri štampanju jedne stranice. Pod tim podrazumevamo cenu boje i papira koje taj uređaj koristi. Ovaj pokazatelj dosta varira i značajan je, jer pokazuje koliko košta eksploatacija štampača. Cena ima smisla za korisnike sa zahtevom od preko 10 stranica mesečno. Podrazumevaju se cene materijala na domaćem tržištu.
- Brzina štampe. Osnovna determinanta brzine jeste tehnologija. Jedinica mere može biti *ppm* (*pages per minute* - stranica u minuti) ili *mpp* (*minutes per page* - minuta po stranici). Naravno, prva mera ide u prilog uređaju, jer i vreme je novac. Na primer, ne može se izjednačiti uređaj koji štampa 20 minuta jednu stranicu sa onim koji štampa dve u minuti, čak i da su im ostali pokazatelji slični. Za osnovu se uzima stranica formata A4 u punom koloru.
- Kvalitet otiska. On se ogleda u rezoluciji štampe, pokrivenosti spektra boja (ruža boja) i tačke, zavisi od vrste tehnologije koja se koristi. Ruža boja je noćna mora za sve kolor uređaje današnjeg vremena. Reč je o mogućnosti da se dobije određena boja. Predstavlja se poliedrom ucrtanim unutar kruga (u krugu je ucrtan spektar boja). Krug predstavlja granicu do koje uređaj može prikazati određenu boju unutar spektra. Što je veća površina poliedra, to je pokrivenost spektra veća, a samim tim raste i kvalitet otiska (slika 1). Ovo je pojednostavljeni oblik, jer ruža boja ima i svoju 3D interpretaciju (Z-osa predstavlja promenu od bele ka crnoj boji), koja nalikuje na nepravilnu čigru posmatranu odozgo.

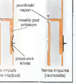
Tri osnovne (primarne) boje, u slučaju kolor štampača, su *CMY* (cijan, magenta i žuta). Kombinovanjem primarnih boja prave se sekundarne (crvena, zelena, plava, itd.). Ostale nijanse iz spektra, u najvećem broju slučajeva, simuliraju se pomoću sekundarnih. Crna boja se nanosi zasebno



Slika 1. Ruža boja

(*black substitution*) ili se dobija mešanjem sve tri osnovne boje. Da bi se dobila kvalitetna crna, potrebno ju je imati odvojeno kao crno mastilo, vosak ili prah (u zavisnosti od tehnologije koja se koristi). Kada se pravi mešanjem *CMY*, tada ona obično naginje prema teget, a na ivicama se naziru osnovne boje (što posebno smeta ukoliko koristimo tekst).

Način pripreme podataka za štampu, podrška raznim kompjuterskim standardima, stručna teh



Slika 2. Princip rada Inkjet tehnologije

...podrška, programska podrška i procesor na kojem se bazira štampač samo su neki od pokazatelja koji se odnose na različite modele unutar jedne tehnologije.

Inkjet

Inkjet (eng. ink – mastilo, jet – mlaz) predstavlja najrasprostranjeniju kolor tehnologiju. Za otisak koristi tečno mastilo koje nanosi direktno na papir.

Princip rada

Piezo-električni kristal, koji se nalazi pored otvora spremnika tečnog mastila, osciluje pod uticajem električnog impulsa vršeći privremeni pritisak na mastilo. U tom momentu kroz otvor izlazi mlaz mastila koji se zaustavlja na papiru formirajući tačkicu (pogledaj sliku 2). Spremnici se nalaze na pokretnoj glavi, čija je funkcija istovetna igličnom štampaču. Kada nekoliko desetina otvora, grupisanih na pokretnim spremnicima osnovnih boja, kontrolisano izbacuju mastilo, dobija se grafički kolor izlaz. Ova tehnologija je veoma stara. Postoji veliki broj modela štampača malih dimenzija sa mogućnošću rada na bateriju. Postoji slična tehnologija (*bubble jet*) koja umesto piezo električnog efekta koristi toplotu za izbacivanje određene količine mastila putem njegovog isparavanja. Osnovni nedostaci tehnologije inkjet-a su razlivanje mastila (slika 3) i sporost.

Pokazatelji

• Cene variraju od ispod 1 000 do 5 000 DEM. Kod jeftinijih modela nema ni govora o ugrađenim fontovima i PostScript-u, već se podaci iz računara prosleđuju kao i kod iglićnog štampača. Problem koji nastaje jeste da je računar blokiran i po više minuta za vreme pripremanja strane za štampu, kao i pri samom štampanju.

• Tehnologija sa najjeftinijim uređajima ima dosta skup otisak. Cena otiska varira u zavisnosti od procenta pokrivenosti stranice bojom. Standardan tekst ima pokrivenost stranice 5%, tekst sa ubačenom fotografijom 10-15% itd. Stranica u punom koloru troši dosta mastila, što je izuzetno skupo (2-3 DEM). Problem razlivanja mastila u ovoj tehnologiji rešava se putem specijalnih papira (pogledaj sliku 3) koji nisu tako jeftini. Presvučeni sjajni papiri i folije koštaju i do 3 DEM po listu. Kada se podvuče crta vidi se da cena jednog kvalitetnog otiska varira od 2 do 6 DEM.

• Brzina štampe je od 5 do 20 *mpp*. Razlog za ovoliku sporost je pomeranje glave štampača više puta preko jednog mesta sa malim pomicanjem papira (da bi se dobila što veća gustina tačaka - rezolucija). Tu npr. *Epson Stylus Color* jedva dobija prolaznu ocenu. Postoji nekoliko modela koji imaju i do četiri puta više otvora za izbacivanje mastila, čime se smanjuje vreme štampe, a postiže se pakovanjem osnovnih boja u fizički odvojene spremnike.

• Visoka prozirnost mastila kao i njegovo razlivanje rezultiraju lošim otiskom (najlošijim od navedenih tehnologija) i to pogotovu kada se štampaju slike. Ni specijalni presvučeni sjajni papiri nisu u stanju da bitno poprave kvalitet otiska. Otisak je podložan spoljnim uticajima, a ponajviše vlazi. Kap vode ili neke druge tečnosti zalutale na kolor otisak trenutno će dovesti do razlivanja boja. Na većini modela jedino je crno mastilo otporno na vodu. Preporuka je da otisak čuvate u fasciklama i vadite ga samo u slučaju potrebe. Za prezentacije koje se izrađuju za grafoskop koristite jače boje.

Idealni korisnici

Kancelarije koje prave nekoliko prezentacija godišnje i to malog obima (gde se boje koriste jedino radi razbijanja monotonije teksta), centri za obuku. Preporučuje se porodicama koje žele da reše problem kućnog štampača.

Inkjet modeli

Hewlett Packard DeskJet 1200C/PS – pristojan kvalitet je osnovna karakteristika ovog modela. *HP* je našao dobru formulu za tržište, što je i rezultiralo uspehom ovog štampača. Daje zadovoljavajuće otiske na specijalnoj foliji. Postiže dovoljan kvalitet i na običnom papiru, što štedi dosta novca. Za razliku od ostalih inkjet štampača, kod njega svaka osnovna boja ima za sebe spremnik, čime se povećava ukupan broj otvora za izbacivanje mastila. To direktno utiče na to da ovaj model štampa znatno brže od ostalih inkjet modela. Ispod dela papira koji se nalazi kod mesta nanošenja mastila nalazi se izduženi grejač koji zagreva papir u trenutku kada se na njega nanosi boja. Time se ubrzava sušenje mastila i smanjuje razlivanje tačaka. Štampač košta oko 5 000 DEM. Postoji veliki broj modela HP-a, Canon-a i drugih proizvođača; njihove cene su do ... DEM, a namenjeni su za kućnu upotrebu ... kao uspešna zamena za iglične štampače.

Solid-ink

Te... mast... stog na sobnoj temperaturi (slično vosku). Ova tehnologija je poznata i pod nazivima *phase change inkjet* ili *hot melt inkjet*.

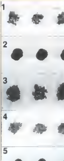
Princip rada

Mastilo se zagreva dok ne dođe u tečno stanje. Zatim se izbacuje, na način sličan inkjet tehnologiji, kroz uske kanale na papir. Pri dodiru sa papirom mastilo se trenutno stvrdne stvarajući tačku. Rezultat ovoga jeste da se može štampati na bilo kojoj vrsti hartije. Otisak je reljefan i može se osetiti pod prstima.

Pokazatelji

• Cene se kreću od 8 000 do 15 000 DEM. Na tržištu postoji samo nekoliko modela koji koriste ovu tehnologiju.

• Trošak je u direktnoj srazmeri sa pokrivenošću stranice bojom. S obzirom na to da može da štampa na svim vrstama hartije koje se mogu naći na tržištu (čak i na običnom papiru, tankoj plastici, pausu, foliji itd.), sam trošak svodi se samo na iskorišćeno voštano mastilo. Za



Slika 3. Razlivanje mastila na raznim vrstama papira
(1 bond paper
2 coated paper
3 ink-jet paper
4 coated ink jet paper
5 special coated ink jet paper)

prosečno popunjenu stranicu on iznosi ispod 0.5 DEM, dok za stranicu sa velikom popunjenošću ne prelazi 1 DEM.

• Brzina je od 2 *ppm* do 5 *mpp*. Kod nekih modela postoji mogućnost da se štampa sa manje prolaza, tako da se probni izlazi mogu dobiti još brže.

• Boje su izvrsne (pune života). Voštano mastilo je neprozirno, tako da boja papira ne utiče na rezultat. To znači da plava boja odštampana na žutom papiru ostaje takva, a ne postaje zelena. Kod štampanja na foliji otisak se dodatno procesira, čime se omogućava dovoljna prozirnost za projektovanje.

Idealni korisnici

Dizajneri proizvoda, umetnici, arhitekte, projektanti i ostali kojima je potreban jak otisak na neuobičajnim vrstama hartije.

Solid-ink modeli

Dataproducts Jolt PSe jedan je od najjeftinijih modela na tržištu. Daje otisak slabog kvaliteta, ali nudi mogućnost štampe na raznim vrstama hartije.

Tektronix Phaser 300i je malo skuplji od prethodnog modela, ali je zato brži i sa znatno kvalitetnijim otiskom. Štampa baš na svemu (autor jednog članka tvrdi da je video otisak na tortilji), robustan je, podržava razne mrežne i pojedinačne sisteme, a ima ugrađen *Adobe PostScript Level 2*, čime rasterećuje računar za vreme štampe. Ovaj model je poboljšana verzija modela *Phaser III*. To se ogleda u dva puta većoj brzini štampe, kao i širim mrežnim mogućnostima. Za stranicu formata A4 u punom koloru potrebno mu je manje od 2 min. Ukoliko se koristi *draft* mod, potrebno mu je još manje vremena. Ovaj model omogućava i štampu do produženog A3 formata (305x457 mm), tako da se može upotrebiti i za štampanje plakata, tehničkih šema, za razne druge grafičke izlaze koji zahtevaju veći format papira.

Thermal-Wax

Thermal Wax (od eng. *thermal* - toplotni, *wax* - vosak) tehnologija našla je praktičnu primenu samo kod kolor štampača.

Princip rada

Rolna folije, sa naizmenično nanesenim tankim slojem voska u cijan, magenta, žutoj i (ponekad) crnoj boji (veličina površine boje na foliji jednaka je veličini A4 stranice) provlači se ispod izdužene glave sa uskim i strogo usmerenim izvorima toplote na mestu dodira folije sa papirom. U trenutku delovanja toplotom na foliju boja prelazi na specijalni papir (pogledaj sliku 4). Papir prolazi tri ili četiri puta ispod glave i na njega se svaki put nanosi po jedna od osnovnih boja. Toplotna glava sastoji se od nekoliko stotina, pa čak i hiljada odvojeno kontrolisanih elemenata, što ovu tehnologiju čini neznatno sporijom od kolor laser štampača. Rolna na koju je namotana folija sa voštanim slojem naziva se transfer rolna. Ova tehnologija je takođe veoma tiha.

Pokazatelji

• Od 10.000 do 20.000 Dem (pretpostavlja ugrađenu *PostScript* opciju, kao i dovoljno radne memorije, odnosno brz procesor u samom štampaču). Kod ove tehnologije elementi moraju biti kvalitetni. *QMS* je pokušao da spusti kvalitet komponenti na svom modelu *ColorScript 230* da bi postigao nižu cenu na tržištu, što je rezultiralo zaglavljivanjem papira, lošim otiskom kao i veoma komplikovanim radom na uređaju.

• Ima fiksne troškove po stranici (od 1 do 3 DEM, u zavisnosti od proizvođača). Cena specijalnog papira koji se koristi je zanemarljiva.

• Najbrži modeli štampaju i do 2 *ppm*, dok ostali neznatno zaostaju (do 1 *ppm*).

• Kvalitet otiska je vrlo dobar, pogotovo kod modela sa većom rezolucijom. Boje su jake i kvalitetne. Može se očekivati dalje napredovanje kvaliteta otiska unutar ove tehnologije.

Idealni korisnici

Kancelarije koje prave veliki količinu prezentacionog materijala, preduzeća sa aktivnim marketingom i uopšte široka poslovna upotreba.

Thermal-Wax modeli

Tektronix Phaser 220. Po testovima brojnih američkih časopisa (*Byte*, *MacUser* itd.) dobio je najviše ocene, i to kao najbolji štampač za poslovnu upotrebu, uz argumentaciju da, kada se uzmu u obzir njegova cena, brzina, kvalitet izlaza, ekonomičnost (ispod 1.5 DEM po strani), pouzdanost, jednostavnost upotrebe (na sebi nema tastera) i ostale mogućnosti, jednostavno nema konkurenciju. Od svih modela koji su pomenuti u ovom članku (izuzev kolor lasera), ovaj model je daleko najbrži. Štampa 2 *ppm* u punom koloru i to u rezoluciji 600x300. To može da zahvali kombinaciji brzog procesora (baziranog na 24 MHz *RISC* procesoru), brze aparature za štampu kao i mogućnosti da koristi *Ethernet*. Ovaj model je dvostruko znan, i to kao trkački i kao zaprežni konj, zbog svoje robusnosti. Čak postoji mogućnost korišćenja običnog fotokopir papira. Maksimalan format papira za štampu je produženi A4 format. Sve u svemu, najbolji odnos kvalitet+performanse i cene.

Dye-Sub

Dye Sub, tj. *dye sublimation* je tehnologija koja obezbeđuje vrhunski kvalitet otiska. Sublimacija je naučni termin koji označava promenu agregatnog stanja direktno iz čvrstog u gasovito, bez prolaska kroz tečno.

Princip rada

Rolna plastične folije, sa naizmenično nanesenim tankim slojem voska u cijan, magenta, žutoj i (ponekad) crnoj boji (veličina površine boje na foliji jednaka je veličini A4 stranice), provlači se ispod izdužene glave sa uskim i strogo usmerenim izvorima toplote (ima ih na hiljade) čija se temperatura, za razliku od glava *thermal-wax* štampača, može precizno i odvojeno kontrolisati. Specijalni papir u ovom slučaju malo je udaljen od rolne, tako da se na njega nanosi uparena boja sa rolne. Za razliku od raznih simulacija boja putem šeme rasporeda tačaka, ova tehnologija koristi promenljivu temperaturu za kontrolisano topljenje tačno određene količine boje (kada se pogleda izbliza nema šarenih tačaka). Svaka ćelija na termalnoj glavi u stanju je da proizvede 256 različitih temperatura. Što je temperatura veća, to je isparenje boje sa plastične transfer rolne veće (da ne bude zabune: papir je udaljen od rolne tek toliko da je direktno ne dodiruje, a boja je u gasovitom stanju samo na delić sekunde). Tako, menjajući količinu boje koja se nanosi na papir, dobijamo savršen izlaz. Ova tehnika zove se *continues tone* (usklađeni ton). Može se štampati samo na specijalnom papiru (providnoj foliji) koji prihvata gasovitu boju u kontaktu sa njim.

Pokazatelji

• Zadržite dah. Od 20.000 do 50.000 DEM. Izuzeci od ovoga su samo pojedini jeftini modeli koji za osnovu koriste ovu tehnologiju, ali ne daju ni približno iste rezultate.

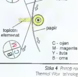
• Troškovi po stranici su fiksni. Iznose oko 6 DEM (u zavisnosti od modela).

• Brzina dosta varira, i to u opsegu od 3 do nekoliko desetina minuta po stranici. Razlog za ovoliku sporost leži u komplikovanosti priprema grafičkog fajla za štampu. Vreme čistog štampanja nije bitan podatak, pa nije ni naveden. Ukoliko štampač ima svoj procesor i memoriju, onda će se računar u velikoj meri, rasteretiti za vreme štampanja.

• Kvalitet otiska je izvrstan. Kvalitet teksta zaostaje za kvalitetom fotografija, ali onome ko ima neki dobar *dye-sub* model neće padati na pamet da štampa mnogo teksta.

Idealni korisnici

Grafički studiji, kartografi, dizajneri (sve vrste dizajna) i ostali kojima je potreban ovako kvalitetan otisak.



Slika 4. Princip rada Thermal Wax tehnologije



Slika 5. Isti zadatak, izlaz u srazmeri 1:1 na raznim modelima istih različitih tehnologija

Redom:

600x300 thermal wax

300x300 thermal-wax

300x300 phase-change (prvi model)

300x300 phase change (drugi model)

300x300 dye-sublimation

Dye-Sub modeli

Tektronix Phaser IISDX od svog izlaska na tržište pobedio je na svim uporednim testovima za najkvalitetniji grafički izlaz. Ovo je prvi model koji je uspeo da verno prikaže osetljive nijanse kao što su npr. boja kože, pesak, stena i sve što vam padne na pamet. Pored kvaliteta karakteriše ga velika brzina štampe (3 mpp). Štampač preuzima *Color PostScript (Adobe Level 2)* fajl od kompjutera, procesira ga na svom 24 MHz *RISC* procesoru tako da maksimalno rasterećuje računar (podsetimo da kod drugih modela računar vrši sve proračune i šalje pripremljene podatke koji se koriste u štampi, čime se blokira računar i po više desetina minuta). Ima ugrađenu podršku za mrežu (ukoliko je na njega priključeno nekoliko kompjutera, on vrši automatsko prebacivanje sa jednog na drugi po završetku posla). Dok štampa jednu sliku, već procesira i drugu. Rezolucija mu je 300 dpi. Njegova optimalna radna memorija je 32 Mb, a cena je dosta visoka.

Fargo Primera svojom izuzetno niskom cenom nudi mogućnost štampanja u *thermal wax* tehnologiji, a sa jeftinim dodatkom i u ovoj tehnologiji. To je sve što nudi. Spor je, boje su loše, a rezolucija je 203 dpi ("koliko para toliko i muzike").

Kolor laser

Kolor laser tehnologija tek je počela da se razvija, a već je postigla zadovoljavajuće rezultate.

Princip rada

U četiri odvojena procesa laserski zrak usmeren na fotoelektrični pojas formira predele sa električnim punjenjem koji predstavljaju mesta za boju koja se trenutno procesira (cijan, magenta, žuti i crni toner). Toner se zatim prihvata na elektrostatička mesta pojasa i prebacuje na valjak koji je pod naponom. Kada se sve četiri boje nađu na valjku, formirana slika se odatle konačno prebacuje na papir, na kojem se presuje i izbacuje iz štampača. Prednost ove tehnologije u odnosu na druge je praktično integrisan klasičan laserski štampač: niski troškovi po stranici, kao i velika brzina štampanja.

Pokazatelji

- Od 20.000 do 30.000 DEM
- Cena je promenljiva, ali je najniža u odnosu na ostale tehnologije. Nije potreban specijalan papir, a otisak je vrlo jeftin - košta ispod 1 DEM.
- Brzina oko 2 *ppm*, a pojedini modeli idu do 3,5 *ppm*.
- Kvalitet teksta je visok i zavisi od rezolucije. Međutim, retki su modeli koji daju zadovoljavajući kvalitet slika i prelaza tonova (samo 600x600 sa *continues tone* tehnikom).

Idealni korisnici

Kancelarije koje imaju zahteve za obogaćenjem klasičnog teksta sa nefotografskim elementima. Samo modeli sa ugrađenim *continues tone* sistemom mogu konkurisati kvalitetnim otiskom fotografija.

Kolor laser modeli

HP Color LaserJet Printer - krajem prošle godine *HP* je izbacio svoj prvi kolor laser štampač pod navedenim imenom. Štampač postiže rezoluciju 300x300. Brzina štampe iznosi 10 *ppm* za crno-bele i 2 *ppm* za kolor dokumente na papiru. Na foliji štampa duplo sporije. Koristi standardne vrste hartije (kao i kopir uređaji). Koristi *dithering* za simuliranje nijansi. Kao prvi desktop kolor laser štampač predstavlja novu stepenicu u razvoju ove tehnologije.

Tektronix Phaser 540 - najnoviji desktop model kolor laser štampača. Nudi rezoluciju od 600x600 dpi. Kada se tome dodaju i korišćenje *continues tone* (fotografije izgledaju mnogo lepše nego kod *dithering* simulacije boja), kao i brzina od 3,5 kolor stranica u minuti, jasno je da je ovo trenutno najjači kolor laser na komercijalnom tržištu. Konstruktori uređaja predvideli su i mogućnost direktnog povezivanja desktop skenera na štampač, čime dobijate i kolor fotokopir. Otisak košta ispod 0,5 DEM. Podržava većinu mrežnih standarda, kao i standarde za *PC* i *Mac*. Kada je povezan sa više sistema paralelno, po završetku primanja sa jednog vrši automatsko prebacivanje na drugi. U stanju je da procesira jedan fajl za vreme štampanja drugog. Radna memorija mu je 32 Mb, a pokreće ga brzi 32 MHz *RISC* procesor.

Još nešto...

Pored štampača koji podržavaju standardne formate javljaju se i štampači koji podržavaju nestandardne, a koriste gore navedene tehnologije. Namena im je specifična, tu srećemo razne štampače kao što su *inkjet* ploteri, uređaji za štampanje *ID* kartica, ulični automati za pravljenje vizit kartica i kolor čestitki sa ugrađenim štampačima itd.

Kada se investira u kvalitetan i skup kolor štampač, veoma je bitno da je obezbeđena stručna i tehnička podrška (autorizovani servis) u našoj zemlji, što većina prodavaca nije u stanju da ponudi, kao i mogućnost direktne nabavke potrošnog materijala. Takođe je bitno da uz opremu dobijete kompletnu literaturu, kao i diskete sa drajverima za sisteme na kojima radite. ■



Skeniranje bez po "muke"

Poslednjih par godina u svetu je u modi "kancelarija bez papira". Elektronski dokumenti bez problema putuju mrežom, ali kako ih dovesti u taj oblik ako vaš koraspodent još uvek koristi klasični oblik komunikacije. Možda je jedno od rešenja PaperPort

Branko Nikitović

Koliko god da je prenošenje rezultata rada sa računara na papir rasprostranjeno putem štampača ili plotera, naravno često treba izvesti i "obrnut" postupak. Verovatno nema korisnika koji bar nekoliko puta nije imao potrebu za skeniranjem. Bez obzira na to da li je reč o tekstu, grafikonima, slikama, slajdovima ili nečem sasvim trećem, potreban je skener. Za većinu korisnika koji svoje računare koriste za obavljanje posla, najčešće se radi baš o unosu samog teksta: faksova, ugovora, ponuda, pisama. Zašto praviti gužvu po stolu i ormanima takvim stvarima – pravo rešenje je preneti ih u računar i na odgovarajući način organizovati.

Kod nas su manje-više rasprostranjeni ručni (*hend held*) skeneri, koji toj nameni baš i nisu dorasli. Ako se radi o A4 stranici, da bi je skenerom prebacili u bit-mapu čitljivu OCR programima, potrebno je izvršiti najmanje dva prolaza, posle kojih dolazi "lepljenje šlajfni", što stvarno nije ni brz ni lak posao. Kažem najmanje dva, jer je poseban problem ravnomerno vući skener preko papira tako da ne dođe do bočnih deformacija – nije neobično višestruko ponavljanje iste operacije. Pored ovoga, na ručnim skenerima je brzina uglavnom vrlo mala, tako da za samo nekoliko strana "ima da osedite". O kvalitetu i načinu ovakvog rada dovoljno govori činjenica da jaki OCR programi na tržištu uopšte ni nemaju drajvere za direktno skeniranje pomoću ručnih uređaja.

Jasno, najpovoljnije rešenje predstavlja stoni (*flatbed*) skener, no njega ne može sebi priuštiti svako. Ako se radi samo (ili najviše) prenos teksta, nije loša ideja ni uzimanje mono varijante, koja je osetno jeftinija. Sa ovakvim mašinama produktivnost je velika, saradnja između OCR programa i boljih modela na tržištu je u potpunosti automatizovana, brzina je veća od bilo kojeg daktilografa.

Programi kao što je *Recognita* i njoj slični imaju izuzetno veliki procenat tačnosti (verovali ili ne, skoro 100% za normalne tekstove iz novina, časopisa ili onih iz pisaće mašine, ako je kvalitet skeniranja dobar), a prepoznaju i naša slova i mogu se podučavati (u šta se svakako isplati uložiti malo vremena).

Posebnu pogodnost za arhiviranje velikih količina teksta predstavljaju dodaci za automatsko uvlačenje listova papira (*auto sheet feeder*-i), koji u sprezi sa skenerom i programom za OCR rade prilično nezavisno od korisnika – slično štampačima. (Opštinske arhive bi sigurno trebalo da razmotre nabavku ovakvih sprava.)

Mana *flatbed* skenera je njihova (za naše uslove) visoka cena, koja ipak nije isplativa za većinu korisnika sa malom kućnom i kancelarijskom primenom računara (*Hewlett Packard ScanJet 3c*, kod nas trenutno najpopularniji kolor skener, u Beogradu košta preko 2500 DEM, dok je njegova mono varijanta jeftinija za nekih 1000 DEM, ali se ne nalazi baš lako).

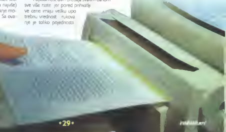
Dakle, šta nam ostaje? Kao i uvek, neki kompromis. U ovom slučaju to je A4 skener koji je fizički tako mali da bez problema stane između tastature i monitora stonog računara, a po ceni u budžet za proširivanje konfiguracije. Postoji više firmi koje prave ovakve "igračke", a najprodavaniji u Americi je po svoj prilici *PaperPort* firme *Visioneer*, čija je cena, preko bare, za novu verziju 2 malo niža od 400 USD.

Popularnost ovih uređaja svakim danom sve više raste, jer pored prihvatljive cene imaju veliku upotrebnu vrednost – rukovanje je toliko pojednostavljeno da praktično i ne postoji (kada je softver već podešen), ne zauzimaju mnogo mesta na gotovo uvek pretrpanim radnim stolovima, imaju brz i tih rad, kvalitet skeniranja je vrlo dobar. Čak je *Delrina*, proizvođač najpopularnijeg programa za faksovanje, počela da prodaje sličan uređaj pod svojim imenom, a i čuveni *UMAX* pravi sličnu mašinu. Adut koji su imali mnogi korisnici faks mašina u odnosu na korišćenje faksmodema bio je da se ne može poslati baš svakako nekom proslediti ponudu ili pismo koje postoji samo na papiru? Sada je i to postalo moguće – i to jednostavnije nego sa stonim faksom. Ako ustreba, sprega skener-računar-štampač postaje i kopir mašina manjeg obima (mislite na toner...), koja još ima i povelike mogućnosti obrade materijala.

Korišćenje kao za decu

Hardverski zahtevi su prilično skromni - minimalno je potreban 386 računar sa bar 4 MB RAM-a (ako ćete primenjivati OCR onda 8 MB) i instaliran Windows softver. Veza sa računarom je serijska, običnim kablom (može se koristiti i D9 i D25 konektor). Pošto nema nikakve interfejs kartice (koja je inače uobičajena za skenere – i stone i ručne), *PaperPort* se može bez ikakvih problema koristiti i sa prenosnim računarima. Softver je tako napisan da ne koristi hardverske interapte, te se može koristiti čak i sa dva već prisutna serijska uređaja.

Fizičke dimenzije ove "bombone" su zaista impozantno male: širok je 30, dugačak



manje od 8 i visok manje od 10 centimetara. Može da skenira u više modova: *normal, fine* i *photo*. Pod haubom se krije jednobitni mono skener, znači ne postoje nijanse sive boje. Rezolucije koje podržava su 200 i 400 dpi, što je sasvim dovoljno za prepoznavanje svih vrsta tekstova. Ako se izabere opcija *photo*, *Paperport* će u toku samog skeniranja (*on the fly*) simulirati polutonove, ali to ipak nije pogodno za bilo kakvu ozbiljniju primenu, mada je dovoljno za logo Vaše firme. Veličine papira koje može da "proguta" su maksimalno 22 puta 76 cm, a minimalno 5 puta 7.6 centimetara - dakle sve od posetnica do A4 formata (i više).

Dok je uključen, *PaperPort* je stalno u *sleep* modu. Na njemu nema nikakvih tastera, čak se i vrsta skeniranja bira softver[illegible]. Kada gurnete papir, sam se "budi" i startuje softver (koji treba da bude već aktivan, [illegible] ga možete minimizovati da ne opterećuje [illegible]). Nikakav pritisak na bilo kakav ta[illegible] [illegible] potreban! Kada se cela stranica oči[illegible] [illegible] obavlja za manje od 10 se[illegible] [illegible] opet sam odlazi u "sleep [illegible]". [illegible] neodoljivo podse[illegible] [illegible] svakako je izvanredan, pogotovo [illegible] mogućnost za neku [illegible]

Softverska podrška

Drugi deo [illegible] predstavlja program [illegible] [illegible] računar. Glavne [illegible]sioneer-ovog softvera su veoma [illegible] organizacije skeniranih [illegible] saradnja sa drugim programima koje imate već instalirane.

PaperPort je proizvod koji bi trebalo da vas oslobodi papirne gužve po stolovima i ormanima, pa je organizacija fajlova od najvećeg značaja. Ne biste dobili puno ako biste umesto loma na stolu imali lom na računaru. Pisci softvera koji stiže uz skener su to očigledno imali na umu i stoga napravili vrlo logičan sistem fascikli (*folders*), koji je uz to i jednostavan za korišćenje. Povuci-i-pusti metod (*drag and drop*) iskorišćen je gde god je to bilo moguće, tako da se npr. štampanje obavlja odvlačenjem ikone dokumenta na ikonu štampača.

Svi skenirani dokumenti čuvaju se u internom *filing system*-u. Ovaj način omogućava brzo i lako pretraživanje materijala, bez obzira na sadržaj koji se prikazuje u obliku umanjenih sličica koje svakako olakšavaju prepoznavanje. Sa leve strane nalazi se lista fascikli (*folders*), a sa desne desktop na kojem se smeštaju sličice. Naravno, prebacivanje između fascikli i radne površine može se izvoditi vučenjem mišem. Ukoliko imate dokument od nekoliko strana, ispod ikone će se pojaviti i mali skrol-bar sa njihovim brojem i trenutnom pozicijom. Za razliku od klasičnog *DOS/Windows* okruženja, možete davati duga imena (slično mogućnostima novog *Windows-a 95*), što je svakako dobrodošla mogućnost. Kada baza skeniranih fajlova postane velika, u pomoć će priskočiti ključne reči, datumi i naslovi, a samim "slikama" dokumenata se jednostavno mogu dodati note, linije i strelice.

Pošto slike zauzimaju mnogo više prostora na disku od samog teksta, verovatno ćete većinu dokumenata prevesti u editabilnu formu. Uz skener se isporučuje *WordScan* firme *Calera*, koji svoj posao obavlja sasvim pristojno. Čak će i veličina slova i formatiranje biti očuvani u slučaju da se tekst prebacuje u *Word for Windows*. Program poznaje zoniranje i autozoniranje oblasti slike, što doprinosi brzini rada i umanjuje kasniju obradu "repova" iz susednih stubaca (ako se skeniraju članci iz novina na primer).

Jedan od razloga prednosti *PaperPort*-a u odnosu na konkurenciju je njegova mogućnost povezivanja sa ostalim programima koje već posedujete. Pri instalaciji, disk se pretražuje u potrazi za poznatim aplikacijama koje se onda automatski registruju i ikone im se dodaju na listu. Tako će (ako je prisutan) *Paintbrush* biti određen za editor slika, [illegible] for *Windows* za obradu teksta, *WordScan* za OCR (prepoznavanje teksta). Sve u svemu, ne može biti lakše.

Aplikacije se pokreću automatski ili kada nad njihovu ikonu dovučete sličicu skeniranog materijala smeštenu u neki od foldera. Spisak podržanog softvera je zaista povelik - preko 50 naslova najpoznatijih proizvođača. Među njima se nalaze *Adobe PhotoShop*, *Microsoft Excel*, *Lotus 1-2-3*, *Novell WordPerfect*, *Lotus AmiPro*, *Delrina Communication Suite*, pa čak i podrška za sve popularniji *IoMega Zip Drive* (eksterni flopi disk kapaciteta 100 MB). Većina potrebnog softvera stiže sa samim *PaperPort*-om, dok se neki linkovi dobijaju uz sam proizvod.

Postoji i mogućnost dogradnje *PaperPort* za radne grupe, pomoću kojeg se dobija jednostavno slanje materijala preko mreže, veza sa *Lotus Notes*-om i sličnim programima, *Lotus cc:Mail*-om i drugima. Podržani su (prema dokumentaciji) e-mail, mrežno faksovanje i grupni rad. Uz ovaj dodatak stiže i *Viewer*, program za pregledanje materijala bez samog *PaperPort*-a, koji se može slobodno distribuirati (mada to kod nas nema mnogo značaja).

Reč za kraj

Šta još reći o ovako usklađenom proizvodu? Proizvođači su obratili pažnju praktično na sve, uključujući tu i ravnotežu hardvera i softvera kao i vezu sa proizvodima drugih softverskih kuća. Korišćenje je toliko jednostavno da se zaista može nazvati "*plug and play*", a praktično nikakva obuka za rad nije potrebna. Jednom rečju - preporučujemo. ■

Autora možete kontaktirati na e-mail adresu brankon@sezam.co.yu



P5/Triton vs 486DX4

Saturn, Merkur, Neptun.. Triton U pitanju su, naravno, imena članova Intel-ove familije čipsetova za Pentium/PCI Posle naziva tri planete Sunčevog sistema, dolazi ime malog jezgra atoma tricijuma - da li to znači da je došlo i do smanjenja performansi samog čipseta Triton? Smanjene su ipak jedino dimenzije i cena, sve ostalo je unapređeno

Saša Krstić

Triton je čipset projektovan krajem 1994 s namerom da omogući osetno poboljšanje performansi korišćenjem naprednih arhitektura pre svega memorije, a zatim i ostatka sistema Dimenzije su smanjene a i cena je osetno niža Najveća novina je upotreba ultra brzog *burst pipelined* sinhronog keša koji (prema merenjima stranih kolega) u startu donosi ubrzanje od 20 30 % Sam čipset u nedostatku ovog keša koristi i 'običan asinhroni SRAM mada su u tom slučaju razlike između *Triton* a i neke druge ploče manje *Intel* je još na samom početku sklopio sporazum sa firmama *Samsung* i *Micron* o isporukama brzog keša (one su ga tada jedine i proizvodile) međutim izgleda da se malo preračunao *Triton* čipset doživeo je neviđenu popularnost te *Intel* tržištu nije mogao da isporuči traženu količinu ploča sa *pipelined* kešom (tj obezbedi dovoljnu proizvodnju/snabdevanje istim) Kasnije su ovaj keš počeli da proizvode *Sony Hitachi Ramtron* te je tržište koliko-toliko zadovoljeno



U međuvremenu [illegible] se poslužio klasič[illegible] trikom tipa 'drpi pa [illegible] dok to ne urade [illegible] i na tržište izbacio [illegible] tipova matičnih [illegible] *Triton* čipovima [illegible] *Endeavour Mo*[illegible] *Zappa* Samo prva je [illegible] dolazila sa pravim *pipelined* [illegible] je bio i zalemljen) dok su osta[illegible] ili sa klasičnim SRAM om (sa mo[illegible] zamene) ili uopšte bez keša. Na[illegible] je isporučivan samo najvećim [illegible] dugogodišnjim komitentima (či[illegible] m) dok su ostali na neviđeno [illegible] kupaca dobijali *Zappa* (bez ika[illegible] u najboljem slučaju *Endeavo*[illegible] klasičnim SRAM om) Ko se bunio [illegible] obećanje da će mu se keš zameniti čim stignu dovoljne količine (da li vam je ovo poznato?) Strani časopisi puni su pisama čitalaca koji pitaju za savet šta da rade pošto su dobili *Zappu* umesto *Aladdin* a.. Uglavnom dobijaju isti odgovor da budu uporni i zovu svaki dan da se raspituju da li su stigli čipovi Šta je to u *Triton*-u osim podrške za *burst pipelined* sinhroni keš što je dovelo do tolike popularnosti ?

Tu je i podrška za EDO memoriju EDO (*Extended data out*) memorija je tip memorije sasvim sličan klasičnom DRAM u s razlikom u jednom kontrolnom signalu koji omogućava brži rad Inače, ubrzanja nisu ni izbliza tolika kao recimo ona koja nudi *pipelined* keš u odnosu na klasičan SRAM Razlike iznose od 5 do 10% (u zavisnosti od vrste L2 keša) a ruku na srce tolika je i razlika u ceni Izgleda da će proći još dosta vremena pre nego što dobijemo zaista drugačiju memoriju sa nekim radikalnijim izmenama. Sve slične evolucionarne promene su privremeno rešenje a gledano unazad brzina memorije se u poslednjih pet godina nije makla sa mrtve tačke tj sa 60 70 ns U međuvremenu, brzine procesora su enormno porasle te se pred keš memoriju kao jednu vrstu tampon zone postavlja težak zadatak

[illegible] ploče imaju i integrisan EIDE kontroler za hard disk a dva najveća noviteta na ovom polju podrška za PIO 4 mod (120ns) i *bus mastering* su ovde implementirana Naravno priloženi su i potrebni drajveri međutim kod većine ploča stiglih krajem avgusta/početkom septembra nema drajvera za *bus mastering* za *Win 95* Isti se nedavno pojavio i može se preuzeti sa SEZAM-a. Ovde treba napomenuti da ovaj kontroler ne koristi 'ozloglašene' PCI IDE (vidi RI 11) čipove koji dovode do korupcije podataka (RZ 000/PCI640IDE) te da kupci ovih ploča mogu biti mirni po ovom pitanju

Jedina *Intel* ploča koja sadrži pomenuti čip je *Premier/Plato* (*Neptun* čipovi) inače slabo zastupljena na ovim prostorima Možda je bitno napomenuti da je s pojavom *Windows* a 95 dokazano da i korisnici tog OS a mogu spavati mirno po ovom pitanju jer *Win 95* ne spada u grupu "rizičnih" operativnih sistema (kao npr *Win NT OS/2 Win 3.x* sa 32 bitnim pristupom ako se ne koristi 32 bitni drajver kuća *WD MicroHouse-EZ OnTrack*..) Na SEZAM u postoji program koji vrši detekciju loših čipova zove se CTRLTEST.EXE a nalazi se u arhivi RZ1000 (konferencija PC HARD)

OEM Triton

Ukoliko se u vašoj mašini već nalazi *Triton* ploča malo je verovatno da je to pravi *Intel* čistunac Grad je preplavljen OEM pločama sa *Triton* čipovima koje nisu ni videle ni brzi sinhroni *pipelined* keš, ni EDO memoriju Iako je i inače nestašica tih čipova glavni razlog leži u ceni 256K brzog keša košta skoro koliko i cela matična ploča (naravno bez procesora) pa dileri sa jedne strane ne vide računicu a s druge ih mrzi da objašnjavaju kupcima šta je to EDO sinhroni keš itd Ovako im je jednostavnije a ko baš želi maksimum neka kasnije dokupi U velikom broju slučajeva (kao što je bio i sa našom pločom) zamena keša brzim BP nije ni moguća jer za njega ne postoje podnožja (tj postoji mesto na ploči što se dobro vidi ali podnožja nisu zalemljena) Da, dobro ste razumeli Sinhroni keš nije fizički ni sličan asinhronom podnožja su odvojena i dosta često postoje samo za jednu vrstu keša (verovatno zbog uštede) Zajednički je samo TAG SRAM čip i u oba slučaja je popunjen

Šta je upoređeno?

Predstavnik 486 tehnologije bila je matična ploča VL/ISA PB486GRN sa Opti čipovima. Kao što kaže ime ploča je sa VESA local-bus magistralom i ima sve GREEN funkcije Procesor je AMD 486DX4/100 V8B što znači nova PLUS serija sa *write back* šemom internog keša Inače ovi procesori su neznatno skuplji u odnosu na *write thro*

Tabela rezultata

ext*MP	486/100 33*3	486/120 40*3	P5/100 66*1.5	P5/120 60*2
SySinfo 8.0				
CPU speed	215.2	259.5	318	381.5
disk KB/s				
sa drajverom	2931	2931	2927	2941
bez drajvera 1899	2361	2927	2940	
Landmark SPEED 2				
CPU speed	333	401	578	694
FPU speed	813	981	1677	2013
VIDEO char/ms	11843	13653	8402	8260
Coretest KB/s				
bez drajvera				
DTR	2832	3408	11272	8640
sekvenojalno	2864	3456	6416	6410
slučajno	1616	1792	2192	2176
sa drajverom				
DTR	9920	11216	14656	13334
sekvenojalno	6592	6512	6416	6416
slučajno	2223	2224	2192	2176
seek ms	9.9	10.1	0.4	10.4
DHD v2.5 KB/s				
bez drajvera				
DTR	2784	3480	10441	8353
slučajno	1606	1816	2142	2142
sa drajverom				
DTR	9280	10441	13921	11933
slučajno	2198	2198	2142	2142
seek ms TR/TR	2.5	2.5	2.7	2.7
seek ms average	10.5	10.5	0.7	10.9
seek ms random	10.9	10.9	1.2	11.1
seek ms FULL	18.2	18.2	8.0	18.1
PC Bench 8.0				
DOSMARK	847	998	1485	1604
procesor	50.5	61	113	128
video II	5090	5770	4846	4655
disk	1819	1973	2225	2362
3D Studio 4				
CAFETABL.3DS				
render sek	75	62	35	30
Windows 3.11				
Wintech 1.3				
overall	58	65	121	131
word	32	36	6	65
CAD	108	122	224	256
sprsheet	29	31	66	68
paint	62	70	131	136
WinTune 1.0				
CPU MIPS	56	66	108	130
FPU MFLOPS	11	13	22	27
VIDEO Mpix/s	4.1	4.4	8.1	8.9
DISK MB/s	8.1	10.0	14.5	14.9
Winbench 95				
graph Winmark Mpix/s	5.33	5.93	11.2	11.9
disk Winmark KB/s	647	750	902	912
FILE MANAGER COPY				
DOOM2.WAD to C (15.5 MB) sec	15	3	7.5	7.5

ugh braću (V8T), a nude unapređenu šemu L1 keširanja, kao i (što ćemo kasnije videti) "overclocking" na 120 MHz.

Ploča ima WinAMI BIOS sa podrškom za četiri hard-diska, posebno *block* mod, LBA, 32 bit transfer za svaki od njih, u delu CHIPSET SETUP sva moguća podešavanja itd. Prilikom *boot*-a BIOS je uredno prijavio Am486DX4 plus procesor. Sama ploča ima 7 ISA (6·16 + 1·8) slotova (od kojih su tri proširena VESA utičnicama), kao i četiri podnožja za 72-pinsku SIMM memoriju.

Naizgled, najobičnija 486 ploča. Međutim, rekao bih da je jedna od tri najbrže koje sam ikada probao. Jedna je od retkih koja dozvoljava sve režime rada: i 3x33, i 2x50, i 3x40. Da, i 3x40 (ali samo sa V8B procesorom). Tako, sa uključenom **auto config enabled** opcijom u SETUP-u dostiže Norton SI faktor od 259.5, što je idealno (tačno tri puta veći od 486/40 koji je 86.5). Naravno, sve jedinice smo pretvorili u nule, burst šeme postavili na 4-3-3-3 (DRAM) i 2-1-1-1 (SRAM) i tako ovu ploču učinili još bržom.

Međutim, V8B procesor nije hteo da radi u režimu 50x2. Brat V8T tu radi normalno (SI=216), ali zato neće na 120 MHz. Napomena: Neki V8T primerci se "bude" i na 120 MHz, međutim, dolazi do grešaka u dijagnostičkim programima u FPU jedinici (npr. NDIAGS: NPU failed), a i HIMEM.SYS se često buni kod testa memorije. Većina njih neće da uđe čak ni u SETUP. Za stabilan 120 MHz *overclocking* neophodan je V8B procesor.

Tako sam rešio da za poređenje odaberem V8B procesor (između ostalog i zbog WB šeme, da tu bude isti kao Pentium). Na brzini od 100 MHz radi kao 33x3, dok na 120 MHz radi kao 40x3 (drugačije i ne može). Grafika je bila poverena sasvim standardnom rešenju - kartici zasnovanoj na *Cirrus Logic* CL5429 čipu, a kontrola hard diska izvanrednom *QDI Vision* EIDE kontroleru 6580+. Znači, tipičan skup *entry level* komponenti, ali sa preporukama u svojoj klasi.

A Pentium?

Gledajući ploču, stiče se utisak da je sve ovo već druga priča. Jes' da je OEM varijanta, ali izgleda da su se Tajvanci svojski potrudili. Na većini čipova je *Intel*-ov znak, i za divno čudo, ni na jednom ne piše "*Made in Korea*". Ploča nosi oznaku GA-586ATE, dolazi sa opširnim uputstvom u boji, *flash eeprom*-om za brzu promenu sadržaja BIOS-a, i disketom sa *bus mastering* drajverima za DOS, ATAPI CD, *Win NT 3.5* (gle) za *Win'95*, *Novell 3.12*, *4.10*, *SCO UNIX*, OS/2 WARP kao i *Fast-disk* drajverom za *WfWg 3.11*. Sam BIOS je firme *Award* i prilično je svež (avg 95). To je jedan od onih BIOS-a sa PnP opcijom (*Plug and Play*), tj. u njega ne morate ni da ulazite. Sve odredi i podesi sam (disk, flopi, PIO, memoriju, WS, COM-ove, IRQ kod PCI...) i sa PnP opcijama *Windows*-a 95 čini idealnu kombinaciju. U njemu možete i ručno da podesite PIO mod hard diska (ako niste zadovoljni AUTO opcijom) a birate između 0, 1, 2, 3, 4. Integrisani PCI EIDE kontroler podržava i PIO mod 4 za maksimalne performanse kod novijih diskova.

Pored 321 pinskog (*socket 7*) ZIF podnožja, nalazi se i VRM (*Voltage Regulation Module*) podnožje. O ovome malo više reči. Pentium familija radi na 3.3 V, međutim, zbog (i dalje) velike količine oslobođene toplote (i potrošnje), koja pre svega ne prija *notebook*-ovima, novi Pentiumi, tzv. *dual voltage* procesori, rade u dvojnom režimu. Samo jezgro radi na 2.5V, dok svi ostali delovi procesora rade na standardna 3.3V. Ovim se štedi 30% energije, a samim tim je i manja količina oslobođene toplote. Ukoliko imate *dual* procesor, u ovo VRM podnožje treba staviti modul, dok su inače pinovi A6-A7 i B6-B7 pokloplјeni *jumper*-om.

Ploča ima četiri ISA 16-bitna, četiri PCI slota, i 6 podnožja za 72-pinsku SIMM memoriju (znači, tri banke). Sve tri banke mogu biti popunjene samo *single-sided* (4, 16MB) modulima, znači ukupno 96 MB. Ako se koriste *double sided* moduli (8, 32 MB), mogu se koristiti samo prva i treća banka i tada je ukupan kapacitet 128 MB. Na ploči su integrisani i

Tabela internog/eksternog takta za Pentium familiju

P5 MHz	60	66	75	90	100	120	133	150	200
ext*mn	60*1	66*1	50*1 5	60*1 5	66*1,5	60*2	66*2	60*2 5	66*3

I/O portovi sa brzim FIFO 16550 UART kolima kao i flopi kontroler Ploča "prima" do 512 K klasičnog asinhronog keša ili fiksno 256 K brzog sinhronog *pipelined* keša

Na žalost imali smo samo klasičan asinhroni SRAM (doduše 512K ali smo za potrebe testa koristili i 256K) Niko u trenutku pisanja teksta nije nudio ove ploče u maksimalnoj konfiguraciji sa BP kešom kao ni EDO memorijom pa smo morali da se zadovoljimo klasičnim rešenjima Ova ploča inače nije ni mogla da se dogradi (kao što je već objašnjeno) Drugim rečima ako u startu ne dobijete OEM ploču sa BP kešom teško da ćete je kasnije dograditi Grafika je bila poverena PCI *Cirrus Logic* 5434 kartici jer je to ujedno i najjeftinije rešenje (5430 nismo mogli da pronađemo) Ista se u VLB varijanti ne može naći kao što se ne može naći ni 5429 u PCI varijanti

Šta je zajedničko?

Zajedničko je 16 MB DRAM-a 70ns Ti i hard disk *Quantum Fireball* 1 1 GB na kojem se nalazi 700 MB raznih podataka programa i testova pod DOS-om 6 20 i *Windows*-om za radne grupe 3 11 U *Win* je odvojeno 4 MB RAM-a za virtuelni keš, znači uključen 32 bitni pristup datotekama bez SWAP fajla a korišćena rezolucija je 800x600 u 65K boja Od testova za *Windows* korišćeni su *Winbench95 Wintach 1 3* i *WinTune 1 0* a pod DOS-om *PC Bench 8 0, Norton SySinfo 8 0 Coretest* DHD (iz lične radionice) kao i *3D Studio 4 0* za proveru brzine renderinga

Kao ešalon u 3D studiju korišćena je demo slika iz tog paketa pod nazivom "*Cafetabl*" renderovan je donji desni prozor (*user view camera 01*) U *PC Bench 8 0* korišćen je modul DOSMARK u okviru kojeg je računat procesorski video i disk indeks Napominjem da je veličina SMARTDRV-a u svim DOS testovima bila 2 MB kao i da je bio učitan drajver QD6580 SYS (osim tamo gde je napomenuto)

Na *Triton* ploči drajver je faktički ugrađen u BIOS (*block mod* PIO) na disketi se dobije samo *bus mastering* drajver koji je i bio instaliran (osim gde je napomenuto) Da bi se videlo imaju li ovi drajveri stvarnog efekta pod *Windows*-om urađen je i jedan trivijalan test koji može svako od vas da uradi i uporedi Izmerena je brzina kopiranja velikih datoteka a u tu svrhu je iskorišćen fajl veličine 15 5 MB DOOM2 WAD (iz istoimene igre) On je kopiran iz direktorijum\DOOM2 u root C:\ uz popunjenost diska od 3/4 Ukoliko nemate ovaj fajl možete iskoristiti bilo koji slične veličine Operacija je rađena iz *File Manager*-a

Komentar za kraj

Zasebno obe platforme u svojoj klasi predstavljaju sam vrh performansi Skrenuo bih pažnju na izuzetno dobre rezultate ove 486 ploče na 100 MHz u režimu 33*3 gde između ostalog dostiže Si index od do sada neviđenih 215 2 (inače je sa VBT procesorom 198) Znači - *AMD Enhanced* serija pokazala se odlično nevolja je što mnogi dijagnostički testovi ovaj procesor "vide" kao *CYRIX* verovatno zbog WB šeme keša (koju je do sada u 486 familiji imao samo *CYRIX*) Između 100 i 120 MHz ovog procesora postoji u svim procesorskim testovima oko 20% razlike (kolika je i razlika u taktu)

U *PC Bench* paketu procesorska razlika iznosi tačno 20% video oko 15% dok je razlika u brzini diska jedva 6% U drugim disk testovima razlika se smanjuje što je i očekivano jer je drajver prilagodio tajming brzini magistrale (33 prema 40 MHz) Jedino u testovima bez drajvera postoje značajnije razlike gde faktički dolazi do izražaja brzina ISA magistrale (CLK/4 u oba slučaja tj 10 i 8 MHz ali sa uključenim *block* modom u SETUP u)

Kod Pentiuma, bolji rezultati disk podsistema dolaze uglavnom od PIO-4 moda (u kojem je *Fireball* radio više nego perfektno) kao i *bus-mastering* i *fastdisk* drajvera (koji "bace" *Quantum* u *Multi word* DMA 2 mod 16 6 MB/s) Pogledajte samo vreme kopiranja datoteke DOOM2 svega 7 5 sekundi Više od 2 MB/s što je skoro duplo brže od 486 platforme

Za nijansu bolji rezultati penfenja na Pentiumu/100 od Pentiuma/120 potiču od veće brzine PCI magistrale na P5/100 Naime Pentium/100 radi eksterno na 66 MHz pa ovaj takt množi sa 1 5 Na 120 MHz eksterno radi na 60 MHz pa množi sa 2 U prvom slučaju, PCI takt je 33 a u drugom 30 MHz (tj polovina) Otuda tako male razlike u video performansama (u *PC Ben*[illegible] P5/100 čak i brži u video operacijama [illegible] Drugim rečima u globalu razlika između dva Pentiuma nisu 20% [illegible] jedino je u 3D Studiju gde je [illegible] rova procesorske snage [illegible]

Kod 486 ploče stvar je malo drugačija Tu su i takt procesora i takt magistrale povećani za 20% pa se to odrazilo i na rezultate Da je ploča radila u režimu 50x2 umesto 33x3 situacija bi bila slična onoj sa Pentiumom čak i drastičnija (jer bi magistrala bila ubrzana 50%)

P5 vs. 486

Kako stoji stvar između Pentiuma i 486 ? Kao što tabele pokazuju duplo veće ubrzanje sa Pentiumom mogu očekivati korisnici programa koji koriste isključivo snagu procesora i to FPU (numeričke operacije u pokretnom zarezu) CAD 3DStudio matematički programi razni proračuni I *Windows* korisnici će zahvaljujući pre svega izuzetno brzom kontroleru hard diska i dobrom drajveru osetiti primetno poboljšanje (najmanje 60%) što inače nije slučaj kod prelaza na Pentium Duplo bolje video performanse su u velikoj meri rezultat bolje grafičke karte (CL5434 64-bitni DRAM bus) U čistom DOS u prosečno ubrzanje pri prelazu na P5 je oko 50% (takav je i moj subjektivni utisak) Obratite pažnju na *PC Bench* video skor CL5429 u DOS-u pokazuje bolje performanse na 486 od bolje kartice na Pentiumu što je potvrdio i *Landmark Speed* test (video brzina)

Što se tiče razlike u ceni, tu 486 platforma ima povoljniji odnos Slična 486 ploča košta oko 370 DEM dok Pentium košta oko 820 dva i po puta više Tamo gde je ubrzanje duplo razlika je isplativa Možda bih ipak savetovao da sačekate još malo - biće još manja 486 više nema kud jeftini je dok se s druge strane pojavio P6-Pentium Pro što će dovesti do daljeg pada cene Pentiuma

Overclocking?

Oba procesora odlično su podnela *overclocking* Ni u jednom trenutku nije došlo do nestabilnosti/greške AMD486plus se odmah probudio na 120 MHz prošao sve dijagnostike čak je istrpeo instalaciju *Windows* a 95 bez ijedne greške Što se tiče Pentiuma on nije hteo da se probudi na 133 MHz (66x2) pa ni na [illegible] 25 kao 50x2 5) Izgleda da je nova Pentium serija znatno osetljivija na *over*[illegible] od stare Ovom Pentiumu/100 je [illegible] 2 [illegible] sećamo se prošle ture Pentiuma 75 koji su mirne duše radili na 120) [illegible] priče da će u skoroj budućnosti Pentium procesori biti fiksirani za jedan takt i jedan množilac čime će se stati na put falsifi[illegible] So overkloken požurite ■

[illegible] možete *kontaktirati na e mail* [illegible] sezam yu



SiliconGraphics
Computer Systems
SiGNET
Complete Solution
Networking & Network Servers
CAD/CAM/CAE
Mr. Si of Graf
SiGNET SiGNET SiGNET SiGNET
– RISC 64bit R8000/R4400
– XL, XZ, Impact, Reality Engine Graphics
Communication Equipment
OEM & Resellers welcome!

Amerikanac i Srbin

Godinama smo se žalili kako Windows 3.1 ima gomilu nedostataka. Danas, međutim, ima onih koji bi radije ostali na toj verziji, nego da pređu na "95-icu". S druge strane je ostao, već otpisani, OS/2 Warp. Da li je on spreman da se pridruži Prletu i Tihom i da zada još jedan udarac neprijatelju?

Boban Petković

Kakve veze ima naslov sa operativnim sistemima? Naizgled nikakve. Ali ako bismo pogledali prosečnog Amerikanca primetili bismo da se on oseća superiorno nad ostalim nacijama, pošto iza njega stoji velika imperija. Voli jednostavnost, pa je tako i sa stanovišta vlasti veoma jednostavno upravljati njime. Realno gledajući, on je daleko od savršenstva. "Truli kapitalizam" učinio je da se i u njemu nađe na nešto truleži, pošto je važnije biti snalažljiv (šta god to značilo) nego biti stvarno dobar ili visoko moralan. Ne liči li vam sve ovo na *Windows*?

S druge strane je jedan prosečan Srbin, možda baš čitalac ovog članka. Da je preterano vredan, ne može se baš reći. Do sada ga je bilo veoma teško naterati da radi, ali u poslednje vreme čini čuda i za malo para. Doduše, upravljati njime nije sasvim lako. Stalno se žali na vlast, kojoj je potrebno prilično iskustva da ga obuzda. Ali kad u tome uspe, on je u stanju da trpi izuzetno loše uslove života, kuću punu izbeglica, dok mu pritom struja i plata stižu samo povremeno. Rečju, kada se svojski potrudi: "Nema te sile ni tog svetskog pakta koji može da predvidi i izračuna šta sve srpski seljak može da postigne i ostvari." Prilično *Warp*-astično.

Šljonulo

Poput šne-nokli drugarice Pepce Kardelj (tag Kabare Laneta Gutovića), i prodaja *Windows*-a 95 je ubrzo počela da splašnjava. Razlog se čini prilično apsurdnim, ali se za pad prodaje može okriviti i suviše agresivna reklamna kampanja. Tako se pred samu prezentaciju činilo da se iza celih "Prozora" kriju *Microsoft*-ovi "čarobnjaci", pa i da potpuni laik može da barata njima. Stari softver je trebalo da radi brže, a novi ubrzo da počne da se pojavljuje. O padanju sistema nije bilo ni reči.

Međutim, već nekoliko dana posle 24. avgusta, prodaja je počela da opada. Iako su korisnici u početku bili prilično zadovoljni dobijenim rešenjem, ubrzo su počeli da pristižu i rezultati iz nezavisnih laboratorija koji su tvrdili da stari programi ne idu brže pod novim sistemom, nego da su primećena i znatna usporenja. Zatim je pod lupu došao i sam korisnički interfejs. "Start" dugme jeste mnogo praktičnije od starog *Program Manager*-a, ali je zato samo podešavanje sistema prilično složeno. Prosečnom korisniku je veoma teško da dođe do opcije koja mu je potrebna, a i neke operacije koje sistem automatski izvodi mogu biti veoma problematične.

Najgora od svega je nesigurnost celog sistema, na šta su pritužbe počele da stižu nešto kasnije. Ne samo da *Windows 95* ume da padne, nego su se pojavile i žalbe na gubljenje velike količine podataka sa hard diska, za šta je optužen *VFAT*. Pritužbi ima mnogo i većina njih su posledica prevelikog verovanja u EPP, za koji je *Bill Gates* pravi mag. Izgleda da mu je većina Amerikanaca, barem do sada, bespogovorno verovala. I nas same čudi da se ovakve situacije javljaju nakon svih onih silnih beta testiranja. Da vas podsetimo, "Računari" su test jedne od prvih beta verzija *Chicago*-a (ili *Windows*-a *4.0*) objavili pre skoro dve godine. Pogledajte taj tekst i primetićete da su izmene minorne. Zato smatramo da se za dve godine program svakako mogao bolje očistiti.

Nasuprot tome, *OS/2 Warp* je imao sasvim drugu sudbinu od *Windows*-a *95*. Njegovi prethodnici bili su nešto popularniji od *Windows*-a *NT*, ali je u svim verzijama prodato samo 30 000 000 primeraka, što je daleko manje od *Windows*-a 3.1x, koji se nalazi (u legalnom izdanju) na preko 50 miliona PC računara. Kašnjenje *Windows*-a 95 poboljšalo je pozicije *Warp*-a, ali ne i dovoljno. Još uvek vlada oskudica *native* aplikacija, mada je prošlo godinu dana od promocije. I u trenutku kada su se i čelnici *IBM*, pod uticajem snažne reklamne mašinerije, gotovo odrekli *Warp*-a, izašao je *Windows 95*, prilično lošiji od očekivanja. *Warp* ponovo postaje zanimljiv, posebno u verziji *Connect*, koja ima ugrađene *mrežne mogućnosti*, dok je i sam rad donekle ubrzan. Na žalost, ona nije uspela da stigne na naš test, pa smo zato koristili standardnu *OS/2-Win* verziju sa sopstvenim delom za izvršavanje programa koji rade pod *Windows*-om 3.1x.

I mali Žile voli stabilnost

Ono što interesuje najveći broj korisnika je stabilnost ova dva sistema, odnosno na koji način i kako padaju. Moglo bi se samo reći da su oba stabilnija od *Windows*-a 3.11, ali umeju veoma lako da odigraju jedan "*Block-Out*". Problem uglavnom leži u tome što oba sistema podržavaju 16-bitne aplikacije, koje često nisu "najkulturnije," pa zato umeju da ih obore. Kod *Warp*-a se to često završava padom samo te aplikacije, mada nam se nekoliko puta desilo i da blokiraju računar, ali pod vrlo specifičnim uslovima.

DOS programi, a naročito igre, umeju znatno više da ga namuče. Često ga blokiraju bez bilo kakvog upozorenja, ali se zato odmah može odrediti sa kojom sve radi, a sa kojom ne. Mada su igre i zabavni softver u najgorem položaju, postoje problemi i sa "ozbiljnijim" aplikacijama. Tako je rad sa *3D Studio*-m ograničen samo na kreiranje modela, dok je renderisanje neizvodljivo zbog nemogućnosti da se nađe *image buffer*. Međutim, u tom slučaju se ništa strašno ne dešava, bivate samo kulturno obavešteni pa znate gde ste. Što je najlepše, kod *Warp*-a, u slučaju da vam sistem padne, pri sledećem dizanju biće automatski startovane sve aplikacije koje su bile aktivne u vreme "nesreće". No, stabilnost celog sistema je na prilično zavidnom nivou, pa će vam poslednja opcija biti izuzetno retko uočljiva.

Windows 95 se malo primakao *Warp*-u i u dobroj meri uspeva da se drži na nogama, mada je, sportski rečeno, "staklene brade". Dešavalo mu se da se sruši od učitavanja, ili snimanja dokumenta u *WinWord*-u. Istina je da tekst nije bio bolji od ovoga koji upravo čitate, ali to zaista nije razlog da se onesvešćuje. Treba znati da se u radu sa 16-bitnim aplikacijama koristi jedna sistemska virtuelna mašina u čijem se jedinstvenom adresnom prostoru, osim aplikacija, nalaze i sistemske biblioteke, pa zato pad nekog od aktivnih programa izaziva i rušenje sistema.

Sa 32-bitnim softverom situacija je identična kao kod *OS/2 Warp*-a koji ima zasebnu virtuelnu mašinu za svaki program, ali uz to ne pravi razliku da li su oni 16- ili 32-bitni. Zato je *Windows 95* sa novim aplikacijama pisanim baš za njega znatno stabilniji. Kada je radio bez onih starih, i u optimalnim uslovima, nije bilo većih ekscesa. Tu je negde na nivou *Warp*-a, mada je na 4 megabajta svaki posao pomalo riskantan, ali zato ima vremena da se popije i jedna kafica za smirenje dok se učitava pasijans.

Važno je reći da pod DOS prozorom priličan deo programa ne radi, ali je zato tu

čist *MS-DOS 7.0* koji obezbeđuje apsolutnu kompatibilnost. Tako će vam, za razliku od *OS/2*, sav stari softver bespogovorno raditi! Ostaje konstatacija da 16-bitni *Windows* i stari *DOS* programi mnogo češće dovode do grešaka sistema nego kod *OS/2 Warp*-a. Ako ćemo pravo, to je i sasvim normalno - kada se *OS/2* ili *Windows NT* sruše u mreži neke velike firme posledice su mnogo veće nego kada *Windows 95* padne malom Žiletu iz Čortanovaca! Zvuči sasvim logično.

Ozbiljnost pokazuje i sistemska disketa koju kreira *Windows*, u poređenju sa tri diskete *OS/2 Warp*-a. Pre svega, *Windows* sistemska disketa će vas podići samo do DOS-a, dok ćete se sa *Warp*-ovom naći u *OS/2* komandom interpreteru, odnosno u text mode-u. Osim toga, *Warp* na disketama čuva kako *Back-up* program, tako i *Restore*. A ako vam se sruši disk *Windows*-om, na vama ostaje da rešite problem povraćaja podataka, makar imali i *back-up* celog diska. Na žalost, vremenom se ispostavilo da je u pitanju veliki nedostatak, pošto se pokazalo da *VFAT* i nije najsigurniji, pa se sve češće čuju pritužbe da je iz čista mira disk "ubijen" ("k'o zec"). Ne znamo da li je krivica do korisnika ili samog sistema, ali bilo je neophodno da to *Microsoft* nekako predupredi, ili da se makar pripremi za ovakve situacije. *Warp* na ovom polju glatko odnosi pobedu!

Slagarica

Mada se grafička okruženja smatraju vrlo intuitivnim i lakim za rad, ona su daleko od savršenstva, pa je ljudima još uvek potrebno vreme za privikavanje (tu nema onoga "Ja sam sama naučila da čitam..."). Doduše, do osnovnih opcija dolazi se relativno lako i posle samo par minuta rada u nekom od njih, korisnik *Windows*-a *3.11* izvodiće osnovne operacije. Ali do onih sofisticiranih podešavanja biće potrebno prilično vremena i truda. Na žalost, ni uputstva nisu pisana da zadovolje sve potrebne aspekte, što se naročito odnosi na *Warp*. On i po ostalim osobinama iz ove oblasti zaostaje za *Windows*-om 95, što je delimično posledica njegove veće starosti, ali i nedostatka inventivnosti koja je, barem u ovoj oblasti, kod *Microsoft*-ovih stručnjaka gotovo neiscrpna.

Osim što se *Windows* rešio *task*-liste i uveo *taskbar*, tu su i mnoge druge novosti. Help je još bolje organizovan, pa ako ga pripitate za nešto, samo kliknite na dugme sa kosom strelicom i već ste na pola puta do cilja. Time je rad značajno pojednostavljen. Dobijanje sudskog spora sa *Apple*-om omogućilo je i daleko lepši grafički izgled u celini. Tako su uvedeni folderi koji u sebi takođe sadrže foldere. Za razliku od starih ikonica, ove nove su objektno vezane za sadržaj na disku i uklanjanje programa sa desktopa uništava ga i na disku. Ako ste pre za lažno stanje, tu su *short-cuts* koji, poput *Warp*-ovih *shadow*-a i *System*-ovih aliasa, predstavljaju svojevrstan "dosovski" rečeno, *batch file*.

Mada će se ova konstatacija mnogima učiniti kao netačna, pošto ovde nije sadržan nikakav programski jezik (za razliku od *Warp*-ovog *REXX*-a), treba znati da se svaki *short-cut* može drukčije konfigurisati. Probajte tako da napravite prečicu za DOS u koju ćete uključiti opciju *Always Suspend* kod *Background*-a. Ako je ona isključena kod osnovnog programa, onda startujte neku DOS aplikaciju i na jedan i drugi način i primetićete razliku.

Short-cut-ovi se lako prave, pa nije nikakav problem ako koji greškom i izbrišete. Međutim, ukoliko se to desi, tu je *Recycle Bin*, ikona u obliku kante za smeće, čiji otpadni sadržaj možete povratiti ako vam je potreban. Novina je i *drag-and-drop* koji se smatra primetnom novošću, mada je takav sistem kod štampe i brisanja koristio i *GEOS* za *Commodore 64*! Uglavnom, izmene su značajne, ali je malo stvarno originalnih, dok je većina sa *Mac System*-a, ili su samo kozmetika. Ono što je svakako dobro došlo je *Start* dugme (startuje se sa, kao *task-list* *Win3.11* i *Warp*-a, znači **Ctrl+Esc**, ili sa **Win** dugmetom kod novijih tastatura), čime smo se konačno rešili starog *Program Manager*-a. Zbog onih sentimentalnih, *Windows 95* ga ostavlja u direktorijumu, [illegible] možete sami da birate. Po nama je [illegible] rešenje desktopa neuporedivo bolje od starog.

Nasuprot drastičnim promenama u korisničkom interfejsu *Windows*-a 95, *OS/2* se nije mnogo izmenio od svoje poslednje verzije. Moglo bi se reći da je najveća novost *Launch Pad*, što je zapravo neka varijacija *HP*-ovog *DashBoard*-a. Mada su po ostalim elementima izuzetno slični, mora se priznati da je način rada ipak u dobroj meri drukčiji od *Windows*-ovog, što se na prvi pogled ne bi moglo reći. Što se samog dizajna tiče, primećuju se sitne bore, ali bi se one lako mogle ukloniti jednom programerskom operacijom. Treba istaći da su sistemski fontovi u *post-script* formatu, čime se dobija na kvalitetu izgleda desktopa, a ikonice su poređane uredno kao kod *Mac System*-a.

Mi, međutim, više insistiramo na jednostavnosti, pa nas zato i čudi što su mnoge opcije prilično komplikovane za izvođenje, te ih je to neuporedivo lakše izvesti kroz "Prozore", na čistom vazduhu *Windows*-a 95. Mada tutorijal veoma jasno objašnjava osnove *Warp*-a, za dalje će vam biti potrebno snalaženje u dosta komplikovanom help-u. Zapravo, svuda je prisutna nespretnost, što se naročito ogleda i u toliko pominjanoj instalaciji. Tu je, recimo, potrebno da neke diskete upotrebljavate više puta za jednu instalaciju, a slično je i kod uključivanja specijalizovanih drajvera, na primer za video karticu, postupak je prilično komplikovan, pošto je osim dvostrukog umetanja svake diskete potrebno "protrčati" kroz nekoliko podešavanja u *OS/2 System folder*-u.

Međutim, i *Windows 95* ima velik broj opcija do kojih se često teško dolazi. Za većinu korisnika ovo je jednostavnije od preturanja po mnogim sistemskim datotekama. *OS/2* vas u potpunosti ne lišava ni čeprkanja u editoru, što, srećom, nije uvek neophodno, mada je potrebno za maksimalne performanse. *OS/2* bez čeprkanja deluje sigurnije (a često i brže) od *Windows*-a 95.

Mada je *drag-and-drop* podržan i ranije od *OS/2*, sada je brisanje fajla podjednako jednostavno kao kod *Windowsa 95*, ali je zato teško reanimirati umrtvljene podatke, pošto je sve izvedeno tako da bi se u tome jedva snašao i Dejvid Livingston! Kako je i ovde reč o objektno povezanom sistemu, brisanje fajla sa desktopa predstavlja i brisanje sa diska. Srećom, to se prvo odnosi na sam desktop direktorijum, a tek ako baš insistirate i na sam fajl! Ovo je sistem koji će odgovarati mnogima koji upravo prelaze sa *Windows*-a *3.11*.

Bacanje u trošak

Iako je za oba sistema najavljeno da mogu da rade već i na 386-icama sa samo 4MB, reč je o čistoj reklami, pošto raditi i "raditi" nikako nije isto. Doduše, *OS/2* pomalo "vadi" postojanje komandnog tekst moda. Mada grafički *WorkPlace Shell* (*WPS*) izgleda veoma lepo, korisnicima sa slabijim hardverom dobro će doći da se reše slika i vrate u čist tekst. Time se značajno dobija na brzini, pa je i rad na 4 MB sasvim prihvatljiv! Što je najlepše, sa *PM API*-jem on je i dalje u stanju da radi sa svim aplikacijama, osim sa onim koje se pozivaju na sam *WPS*. Rad i dalje nije brz koliko treba, već bi vam za to bila potrebna jedna solidna 486-ica čiju snagu *OS/2* u dobroj meri ume da iskoristi. Pohvalno je što je od jednog sistema, gladnog memorije i procesorske snage, vremenom nastao po zahtevima u suštini ipak skroman OS.

Nasuprot njemu, *Windows 95* čini se kao mala ala. I sama reklamna kampanja imala je slogan "Da li ste spremni za *Win95*?"! Šta dalje očekivati?! Istina je da oba sistema u grafičkom modu rade podjednako na 386 sa 4 MB (što bi se teško moglo nazvati radom), za pristojan "život" 95-ici je potrebno još oko 12 MB i po mogućstvu 486-ica. Nasuprot njoj, *OS/2* više insistira na svojoj procesorskoj snazi, a znatno manje na memoriji, što je izuzetno važno danas kada je memorija i dalje veoma skupa (delimično

Windows 95 vs Warp

i zbog *Windows-a 95*), dok procesori gotovo svakodnevno pojeftinjuju. Za njega je i 8 MB sasvim dobro, ali je zato veoma poželjna jedna 486/100 mašina. Preko 16 MB oba sistema ne pokazuju neko naročito ubrzanje, pa zato ne razmišljajte o tome, osim ako se bavite izuzetno kompleksnim programima, recimo za obradu slika. Slično je i sa procesorom, pošto kod oba snaga Pentium-a još uvek u potpunosti ne dolazi do izražaja.

Brzina rada sa *native* aplikacijama je kod oba sistema približna, mada malu prevagu odnosi *OS/2*. Razlog tome delimično leži i u nešto manjim hardverskim zahtevima, pa je na istoj platformi *Warp* uspevao da radi brže. Takođe, javljaju se i ubrzanja kako u DOS-u, tako i u *Windows* programima pod *OS/2*. Ono što je posebno zanimljivo je to da ubrzanja nisu konstantna, nego zavise od strukture programa, pa ako ovaj baš ne odgovara *Warp-u*, *Windows 95* ga ume potući i to znatno ređe.

Veoma nas je iznenadila i brzina izvršavanja programa sa puno matematičkih proračuna. Tako je na primer jedan kompleksni demo izvršen pod *Warp*-om 40% brže nego pod *DOS*-om 7.0 i pod *DOS* sesijom *Windows-a 95*. Osim toga, *Warp* posle "zagrevanja" uspeva da prestigne i *Windows 95* u radu sa starim 16-bitnim *Windows* aplikacijama. Mada mu prvo podizanje "Prozora" ili neke od aplikacija za verziju 3.1 odnosi oko dva puta više vremena od *Windows-a 95*, kasnije se razlika preokrene u suprotnom smeru, i to znatno. Tako *Warp* uspeva da startuje *Corel 5* za samo 5 sekundi, naspram 15 sekundi koji su potrebni *Windows-u 95* na istoj konfiguraciji! Ako se tu dodaju sigurnost i brzina koje se osete u radu, primećuje se da je *OS/2* zaista zaslužio ime *Warp*! Doduše, rezultati koje navodimo postignuti su na 486DX4-100 sa 20 MB RAM-a, što je u našim uslovima nešto iznad proseka. Mada memorije ima dovoljno za komotan rad *Windows-a 95*, izgleda da kod njega postoje i neki problemi u "genetskom kodu" koji mu ne dozvoljavaju da potrči.

Plavo, volim te plavo

Mada nam se više puta do sada činilo da je *IBM* izgubio smisao za PC tržište, svojim *ThinkPad*-om i *OS/2 Warp*-om ipak je pokazao da još uvek može da se takmiči sa "malom" konkurencijom. Doduše, "plavi" još uvek ne umeju "da podignu prašinu" kao što to rade ljudi iz *Microsoft*-a. Ipak, oni su donekle pomogli i *Warp*-u pošto je stvorena klima da je dvojac *DOS-Windows 3.1* završio karijeru. Ne bismo se složili. Taman kada smo hteli da rezultate testiranja pretočimo u ovaj tekst, *OS/2* je odbio da koristi *Windows* aplikacije (posledica ručnog "čačkanja") a *Windows 95* se toliko uzjogunio da je i sam *WinWord* prešao u ilegalce sa svojim nedozvoljenim operacijama. Pogađate gde je bio izlaz? Srećom, posle smo uspeli oba da pripitomimo, ali nam je za to trebalo vremena.

Na žalost, nismo uspeli da ubedimo ni jedan ni drugi da koriste *Twain* drajver sa skenera. Zato smo skeniranja nastavili da radimo u *W4WG 3.11*, pa je naša mašina bila u stanju da radi sa čak 4 (četiri) operativna sistema (dok se peti, *DOS 6.2*, nalazio u jednom direktorijumu za "ne daj bože"). Važno je reći da *Windows 95* rado ugasi particiju *Boot Manager*-a koji *OS/2* kulturno instalira. Ali zato, ako *Warp* instalirate na butabilnu DOS particiju, vaš *Windows 95* će ostati bez teksta, pošto je sada zavladao **baš** pravi operativni sistem kojem nije potrebna pomoć DOS pri učitavanju. Sve u svemu, trude se da jedan drugom napakoste, ali u suštini mimo rade svaki u svom ćošku (ako ga imaju).

Znači, ako ste konzervativni, posedujete opremu bez potrebnih drajvera ili imate relativno slab hardver, za sada nema previše razloga za prelazak na neki drugi OS. Od najavljene bujice novih programa za *OS/2* i *Windows 95* trenutno se ne vidi mnogo, pa možete još uvek samo da uživate u *Program Manager*-u.

Ipak, oba nova operativna sistema donose i mnoge prednosti. Pre svega rade u *Protected* modu (*Windows*-ima je trebalo 9 godina od nastanka 386-tice!), čime je onemogućeno da neki program koristi oblasti koje su već iskorišćene od strane operativnog sistema. Osim toga, *multitasking* je postao *preemptive*, čime su aplikacije konačno postale nezavisne. *Multithread* nam omogućava da započnemo sa nekim drugim poslom pre završetka prvog. Više nam nisu potrebni ni *memory manager*-i, pošto je njihova funkcija "ravno do Kosova"... vide je i... nearno, a i *plug-and-play* je... osobina. Grafički sistem, osim što je intuitivan, nudi i objektni rad. Razloga za prelazak na novi sistem ima dovoljno, ali koji je pravo rešenje?

Po nama, to je ipak *OS/2* ("Za dlaku šinko") pošto se pokazao kao brži, stabilniji sa podrškom za više DOS konfiguracija virtuelnih mašina. *HPFS* sistem je brži i pouzdaniji od *VFAT*, ali zato ne nudi kompatibilnost. Srećom, *FAT* je podržan, a postoje indicije da će se i za *Windows 95* pojaviti odgovarajući drajver koji će podržati *HPFS*. Sa svojim *Boot Manager*-om omogućava i jednostavno koegzistiranje još nekog OS-a, a naša preporuka bi bila da ipak zadržite i *DOS* ako nameravate da pređete na *Warp*. Osim toga podržava i sisteme sa 4 procesora, pa čak i mrežu u *Connect* varijanti. U osnovnoj verziji jeftiniji je od *Windows-a 95*, što se može reći i za većinu *native* aplikacija. Doduše, *OS/2* za mreže je duplo skuplji, ali on se nalazi na pola puta od *OS/2* ka *Windows-u NT*. Ipak nije u pitanju ista kategorija.

S druge strane, *Windows 95* u startu nudi mrežne mogućnosti, dok se u onom dugom *Start*-u krije ključ njegove jednostavnosti. Help sistem je još bolji od onog što su zvali *Microsoft*-ovim. Postoji more aplikacija, ali sve se mogu izvršavati i pod *Warp*-om. Ipak, ubrzo će se pojaviti i 32-bitne verzije svih poznatijih aplikacija, što se takođe može uzeti kao prednost. No, ako uzmemo u obzir *VFAT*, pomalo sumnjivog kvaliteta, i ugrađen virus (koji skuplja podatke o softveru na disku, da bi ih kasnije poslao *Microsoft Network*-u u tajnosti i doveo vas na sud ako koristite "pirate"), može se reći da je *Microsoft*-ova arogancija dostigla vrhunac (bar za sada) i da im se to ipak može obiti o glavu. Korisnici se sve glasnije žale. Da li će Bili umeti da ih umiri?

Volite li Crnjanskog?

Do sada se nebrojeno puta dešavalo da lošiji proizvod ima bolju prođu na tržištu. Hoće li se priča *VHS*-a i *Video-a 2000* ponoviti? Kako se stvari sada odvijaju, samo polovina korisnika *DOS+Windows* sistema preći će na novi *Windows 95*, čime bi on zauzimao 40% PC kolača. *Warp* zato ide na onaj ostatak, pa bi se lako moglo desiti da umesto svojih 10% zauzme četvrtinu tržišta, čime bi imao veće šanse u budućnosti. Ukoliko bi se pojavljena verzija koja izvršava 32 bitne aplikacije za *Windows 95* pojavila ranije, *OS/2* bi predstavljao veliku pretnju po *Microsoft*.

Ipak, *Windows* se sigurno neće tako lako predati. U krajnjoj liniji, on će još uvek dolaziti preinstaliran na milionima računara, što *IBM* tek sada čini, naročito u Nemačkoj gde je *Warp* najpopularniji. Međutim, ako se uzme u obzir da je kernal *OS/2* uradio *Microsoft* i da se on ne sme menjati, vidi se da je *IBM* u prilično velikom problemima, pošto nije u mogućnosti čak ni da ispravi sitne greške, a kamo li da izvede neka veća unapređenja. Ipak, već se pomalo šuška o naslednku, čime bi se borba nastavila, a od čega bi najviše profitirali korisnici! Ipak, za kraj ostaje pitanje šta vi da odaberete. Pokušaćemo da vam odgovorimo pričom sa početka.

Windows je u svetu izuzetno popularan, ali se ipak nađe neko kao Nemci i naročito Kinezi koji su stali na stranu *Warp*-a. Baš kao što Srbi u svetu nisu baš omiljeni, ali ih tu i tamo neko podržava. Čak i kod nas ima dosta onih koji baš nisu zadovoljni time što su ovde rođeni, pa se sele u neke druge svetove. Međutim, ako volite inat, probleme, ali pre svega dušu (ono što bi trebalo da krasi prosečnog stanovnika ovih krajeva), izaberite *Warp*-a, verujte, nećete se pokajati, makar ceo svet bio protiv vas! ■

Prednost nove generacije

Koliki se značaj pridaje evaluaciji programskih paketa pod operativnim sistemom *Windows 95*, dobru ilustraciju možete videti iz primera poznate američke firme *Kelly Services Inc.* čija je osnovna delatnost regrutovanje, obuka i usavršavanje službenika u kancelarijskim poslovima širom kontinenta. Broj od 325 hiljada komitenata godišnje predstavlja podatak za poštovanje, a za ovu firmu obavezu da čuva svoju reputaciju veoma brižno.

Dragan Grbić

Juna meseca završena je i objavljena detaljna studija o komparativnom testu izvršenom nad dve generacije programa *Microsoft-a* – *Office 4.3* i *Office 95*. Test je obavljen u laboratorijama firme *Kelly* u gradu Troj, Mičigen. Cilj studije su tri grupe informacija: prvo, merenje produktivnosti krajnjih korisnika koja se postiže migracijom sa *Office 4.3* na *Office 95*, drugo, merenje tzv. linije učenja povezane sa ovim prelazom, i treće, identifikacija potrebnog nivoa obuke zaposlenih u sopstvenim redovima. Test se sastojao iz skupa uobičajenih operacija kakve se obavljaju svakodnevno i koje je trebalo izvesti nad obe verzije paketa: pronalaženje, otvaranje i snimanje dokumenata, kreiranje, editovanje i formatiranje dokumenta, analiziranje podataka.

Sam test tekao je u pet faza:

- **Odabir učesnika u testu** - odabrano je 35 dobrovoljaca iz redova firme, iskusnih u radu sa *Office 4.3*. Deo učesnika je radio i test u Access bazi.
- **Test Office 4.3** - od učesnika je zahtevano da izvrše skup zadataka, koristeći *Office 4.3* aplikacije, najveštije i najbrže što mogu. Praćeni su vreme i tačnost obavljanja,
- **Samostalna obuka** - da bi se upoznali sa *Office 95*, učesnicima je dato na raspolaganje 45 minuta da pregledaju *Microsoft Office 95 Guided Tour* materijale za obuku. Ovaj materijal sadrži datoteke sa primerima i instrukcije za samostalno otkrivanje novih mogućnosti paketa. Učesnicima je takođe prikazana petnaestominutna video prezentacija programa *Windows 95* i *Office 95*,
- **Test Office 95** - od učesnika je zahtevano da izvrše ekvivalentan skup zadataka kao u prethodnom testu, koristeći program paketa *Office 95*. Praćeni su takođe vreme i tačnost izvršenja.
- **Završna anketa** - učesnici su na kraju zamoljeni da odgovore na nekoliko pitanja kojima daju lično mišljenje o viđenom.

Svi kandidati radili su test na istovetnim računarima firme *Dell* klase Pentium 100 Mhz sa 16 MB RAM. Test *Office 4.3* obavljen je na *Windows 3.1* platformi, dok su u drugom testu upotrebljene Beta verzije *Windows 95* i *Office 95* paketa. Na raspolaganju su bile kompletne baterije Help datoteka svih programa.

Da bi se izbegao uticaj vežbe u toku testiranja, razvijen je veliki broj kratkih upita, koji su potom emitovani korisnicima po slučajnom rasporedu. Rad u *Excel-u* i *Word* u praćen je automatski programom *PinPoint*, koji je firma *Kelly* razvila za sopstvene potrebe obuke i testiranja kandidata. Delovi testa rađeni u *Access-u* i *PowerPoint* u mereni su štopericom. Princip u ocenjivanju bio je da se uvažavao samo kompletan i tačan rezultat. U analizi rezultata testa uzimani su u obzir samo oni učesnici koji su kompletno završili pojedinačne zadatke na obe platforme.

Analiza produktivnosti pokazala je da korisnici značajno brže obavljaju zadatke u *Office 95*: prosečno vreme izvršenja svih zadataka palo je sa 32.28 na 20.4 minuta (vreme je kraće za 37%). U grupi zadataka koji se tiču kreiranja i editovanja, primećeni su sledeći momenti:

- formatiranje teksta u *Word*-u zahteva 45% manje vremena - ovo je direktan rezultat **AutoFormat** opcije, koja funkcioniše interaktivno u toku kucanja teksta,

značajno vreme (oko 30%) uštedi se u adresiranju korespodencije u *Word*-u primenom opcije **Address Book**

- vreme potrebno da se odštampa tekuća *PowerPoint* prezentacija i startuje sledeća palo je sa 131 na samo 62 sekunde
- korisnicima je trebalo 9 sekundi više vremena (23%) da unesu tri reda podataka u radni list u *Excel*-u

TABELA 1	Office 4.3	Office 95
Konzistentnost paketa	4.8	6.0
Lakoca izvršavanja zadataka	5.0	6.0
Mogućnosti u toku izvršenja	4.9	6.1
Preformansa programa	4.7	6.1
Kvalitet i upotrebljivost Help baterije	4.4	5.7
Lakoća učenja	5.1	5.7

U poslovima analize podataka rezultati su bili sledeći:

- vršne vrednosti u skupu podataka u *Excel-u 95* nalažene su za skoro dva minuta manje vremena nego u *Excel-u 5*
- u proseku, 78 sekundi ušteđeno je u identifikaciji stavki koje se mogu kupiti datim iznosom budžeta primenom *Excel-a 95*
- korisniku treba duplo manje vremena da pripremi i izvede model prognoze prodaje u *Excel-u 95* nego koristeći isti model u *Excel-u 5*
- koristeći *Access 95*, korisnicima je trebalo oko dva minuta manje vremena da pronađu komitenta u bazi prema zadatom kriterijumu, nego koristeći *Access 2*.

Analiza tačnosti takođe pokazuje prednost nove generacije programa. Detalji su sledeći:

- svi ispitanici su potpuno ispravno adresirali pisma u *Word-u 95*, naspram samo 59% uspešnih u *Word-u 6*, **Address Book** je ključni artefakt ovog uspeha,
- iako nije bilo razlike u vremenu kreiranja "bulet" liste nakon premeštanja rasporeda podataka, 88% korisnika *Office 95* ovo je uradilo ispravno, nasuprot samo 26% korisnika *Office 4.3*
- 92% korisnika *Excel-a 95* lokalizovalo je vršne vrednosti u tabeli, nasuprot 69% korisnika *Excel-a 5*
- u *Office 95* dvostruko veći broj korisnika uspeo je da identifikuje stavke koje se mogu kupiti datim budžetom

62% korisnika *Word-a 95* bilo je u stanju da uspešno pronađe i otvori datoteke zadate ključnim rečima, naspram samo 21% korisnika *Word-a 6*,

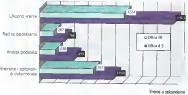
- samo 18% korisnika *Office 95* bilo je u stanju da snimi novu datoteku u novi folder naspram 35% korisnika *Office 4.3*. U *Office 95*, naime, korisnik može da kreira novi folder u dijalogu za snimanje dokumenta. Nakon što bi korisnici kreirali novi folder, zaboravili bi da ga selektuju pre snimanja datoteke.

Korisnici su u anketi zamoljeni da daju ocene od 7 ("vrlo zadovoljan") do 1 ("vrlo

	NE		DA		Neopredeljeni	
	broj	%	broj	%	broj	%
Da li biste mogli efikasno koristiti Office 95 bez dodatne obuke?	5	14%	24	70%	6	16%
Da li biste izvršili upgrade na Office 95?	0	0%	30	86%	5	14%

Vreme izvršenja zadataka

prosek za sve ispitanike



nezadovoljan"). Prosečne ocene vidite u tabeli 1. Dajući ove ocene i odgovarajući na pitanja, ispitanici su bili gotovo jednodušni u odluci: *Office 95* je poželjan naslednik prethodnog paketa.

Možda bismo mogli zameriti da je testiranje, iako pažljivo izvedeno, obavljeno na nedovoljno velikom uzorku, a svakako nedovoljno raznovrsnom. No, treba da bude jasno da je vrlo teško organizovati ovako homogen test nad većom populacijom, a da sa svoje strane preciznost u analizi kompenzuje ovaj nedostatak. Krajnji cilj je postignut, a odgovor dat: *Office 95* ima sve atribute ultimativnog alata u poslovnom okruženju. Odavno postignuta intuitivnost u programima otišla je i korak dalje, što je posledica kvaliteta samog *Windows-a 95* kao platforme i njegovih mogućnosti. Zbog poređenja izvršenja radnih zadataka, uzeta je u obzir kompletna operativnost kakva postoji u *Office 4.3* i zbog toga su noviteti u *Office 95* bili skoro zanemareni u analizi. Novitete, svakako, posmatramo kao čist bonus u ovom slučaju. Ako ove silne navedene brojke, histogrami i tabele mogu da se svedu na jednu rečenicu, onda ona verovatno zvuči kratko: Dobrodošli u *Office 95*! ■

Autora možete kontaktirati na e-mail adresu dr.grba@sezam.co.yu

Literatura

Microsoft ® Office 4.3 versus Office for Windows 95. User Productivity Study - © 1995 Kelly Services Incorporated

	Office 4.3		Office 95		Jednako	
	broj	%	broj	%	broj	%
Koja aplikacija je bila lakša za upotrebu?	3	8%	23	66%	9	26%
Koja aplikacija vam je omogućila da zadatke kompletirate efikasnije i brže?	1	3%	31	89%	3	8%
Koja aplikacija vam se više dopada?	1	3%	30	86%	4	11%

YU Windows 95

Windows 95 Pan European je Microsoft-ovo viđenje naših potreba. Podrška za nas već u standardnoj verziji Windows 95 postoji i za ćirilicu i za latinicu, ali duboko ispod žita. U čemu je razlika? I kako da standardni Windows 95 prilagodite sebi: fontovi, naša slova, natpisi, standardi...

Srđan Pantić

Pojava finalne verzije *Windows 95* donela je što se nas tiče primetno razočarenje. Ne zbog samog operativnog sistema – tu je *Microsoft* napravio sjajan posao – nego zato što je podrška za našu zemlju ispanla iz finalne verzije, za razliku od prethodnih beta i test verzija u kojima smo bili sasvim uredno podržani i sa latinicom i sa ćirilicom. Na dan izlaska Windows 95 na svetsku scenu 24. avgusta 1995. godine izgledalo je da smo ne i prvi put zaboravljeni. *Microsoft* je naime u osnovnoj ("američkoj") verziji *Windows 95* u internacionalnoj podršci podržao veliki broj zemalja grupišući ih u četiri grupe: *USA/Western Europe* (kao osnovnu), *Central European*, *Cyrillic* i *Greek*. U spisku su se našli i Slovenija, Bugarska, Rusija i mnogi drugi koji za *Microsoft* očigledno predstavljaju veoma interesantna tržišta (interesantnija, svakako od našeg, barem u ovom trenutku). Ipak odmah se moglo primetiti da je prisutna podrška za kodne strane 1250 i 1251 u kojima su retrospektivno smeštene srpska latinica i srpska ćirilica. Ipak *Microsoft* je bio najavio izlazak verzije *Windows95 Pan European* koja je namenjena detaljnom pokrivanju evropskog tržišta, te time i nas.

Pan European?

Šta tačno donosi podrška za pojedinu zemlju u *Regionals Setings* iz *Control Panel*-a? Osim jasno podrške nacionalnim alfabetima u *true type* i bitmapiranim fontovima, *Windows 95* donosi i podršku za znake u DOS 7.0 (koja potiče još iz MS DOS 5.0), nazive i datume ispisane na lokalnom jeziku, data je podrška pravilnom sortiranju po lokalnom alfabetu i tačni standardi za tu zemlju kad je reč prikazima datuma, vremena, valute, decimalne tačke/zareza, kalendara i drugog. Automatski se podešava vremenska zona, prilagođava promena vremena ako je zemlja podržala letnje i zimsko vreme (sistem sam vodi računa o pomeranju za jedan sat pri prelasku sa letnjeg na zimsko vreme i obrnuto).

Takva podrška pružena je i nama u okviru evropske verzije *Windows 95 – Pan European*. *Microsoft* nam je pružio podršku za oba pisma – srpsku ćirilicu i srpsku latinicu. Ispravno nam je podržan kalendar (gregorijanski), formati prikaza datuma i vremena, sortiranje po abecedi i azbuci, valuta (dinar, naravno), decimalni zarez i drugo. Ovoga puta *Microsoft* je išao do kraja sa detaljima. "Konačno nema potrebe za raznim budženjima!" bio je prvi komentar mnogih, uključujući i autora ovog teksta. Reklo bi se da je sve sjajno i da problema nema. Da bi se imala kompletna podrška za Jugoslaviju, dovoljno je kupiti *Windows 95 Pan European* umesto standardnih *Windows 95* i svi problemi su rešeni! Na žalost, pogrešno.

Po svemu sudeći mi smo jedna od retkih zemalja koja u upotrebi ima dva pisma – ćirilično i latinično. Pošto ni *Windows 95* nema podršku za *Unicode* (u kojem bi se u okviru jednog fonta nalazila sva pisma u upotrebi na planeti a jedan karakter bio bi predstavljen sa dva bajta) već je i dalje u upotrebi modifikovano rešenje sa različitim fontovima, nastao je problem podrške za srpsku ćirilicu i srpsku latinicu u sistemskim fontovima, pri čemu se engleska latinica nije smela zaobići. Pošto mi ipak nismo toliko važni da bi *Microsoft* kreirao posebne fontove za nas, u *Windows 95 Pan European* kao osnovni izbor dat je *Serbian/Cyrillic*. To znači da nam je u sistemskom fontu prisutna srpska ćirilica i engleska latinica dok naše latinice nema. Imate je naravno u prisutnim *CE True Type* fontovima, ali ako vam ista zatreba u bitmapiranim fontovima možete to slobodno da zaboravite. Zato će svi nazivi (ne i sam *Windows 95*, naravno) dana i meseca biti na srpskoj ćirilici.



Kako stoje stvari sa DOS-om? Ako ste kojim slučajem koristili ranije kodnu stranu 852 (inače baš *Microsoft* ovo rešenje za našu latinicu još od *MS DOS 5.0*), možete da zaboravite na nju. Podršku za nju iz kutije (kao gotovo rešenje) nećete dobiti. To ne znači da je nema, naprotiv, ali izbor je ćirilica, sećate se? Cinici me uveravaju kako je i inače out raditi u *DOS*-u i kako nam to i inače ne treba. Problem je što im nešto slabo verujem.

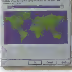
Dakle, *Windows 95 Pan European* pokazuje se kao blag šok za svakoga ko koristi i srpsku latinicu u svakodnevnoj (računarskoj) upotrebi, pogotovo prilikom komunikacija na BBS-ovima i, naravno, SEZAM-u. Ipak za onoga kome ovakvo ćirilično rešenje odgovara, *Windows 95 Pan European* je sjajna stvar i on nema potrebe da čita ostatak članka. Ako vam, pak, treba srpska latinica, nastavite dalje.

Ljubitelji *Windows 95* **standardnih** rešenja pristupaju kompromisima i odlučuju da progovore slovenački. Jednostavno izaberu kao zemlju Sloveniju i postaju novopečeni Janezi, ali sa pozitivnim nuzefektima pojave srpskih latiničnih znakova. Ako izaberete standardni prebacivač tastatura istom prilikom dobićete i standardnu tastaturu koja će vam možda i odgovarati.

Alternativa

Da li mora baš tako? Zar ne postoji način da *Windows 95* legitimno prizna i našu drugu verziju *Serbian/Latinic* koja je, barem što se računara tiče, za najveći deo korisnika ipak potrebnija. To, naravno, ne znači da ćirilica nije potrebna, naprotiv. Ali mnogo češće je potrebno otkucati tekst na ćirilici nego što nam ćirilica treba kao sistemski font u *Windows* programima. Podršku za ćirilične *true type* fontove imamo od starta i tu nema problema.

Dakle, pitanje glasi: kako doći do srpske latinice kada se kao zemlja izabere Srbija? Malo koliko tražili u *Regional Settings* (iz *Control Pa-*



nel-a) u *Windows 95 Pan European* nećete naći ništa drugo do *Serbian/Cyrillic* i ničega u standardnim *Windows 95*. Ipak, nemojte gubiti nadu. Ako se sećate, takva podrška postojala je u beta i test verzijama i - ispostavilo se da ona i dalje postoji.

Dakle, potraga za *Serbian/Latin*-opet nas vodi ka prvoj verziji *Windows 95*, bez obzira na prvobitno razočaranje i njenoj prepravci. Očito je da se "budženja" operativnih sistema skoro nećemo osloboditi. Srećna okolnost je u tome što je ovde reč "budženje" s pravom u navodnicima, jer *Windows 95* ima podršku za nas i po ćirilici i po latinici; jedino što do toga treba doći. Ali, krenimo redom.

True type fontovi

Prilikom instalacije *Windows 95* (ili kasnije) potrebno je da postavite kao izbor *USA* i engleski jezik, odnosno da izbegnete pojavu *Microsoft*-ovog prebacivača tastatura u *Taskbar*-u. Zatim je potrebno da odete u **Control Panel/Add/Remove Programs**, da izaberete *Windows Setup*, da dva puta kliknete na stavku **Multilanguage Support** i da izaberete podršku za centralne evropske i ćirilične jezike (*Central European Language Support* i *Cyrillic Language Support*). Time dobijate standardne TT fontove sa podrškom za kodne strane 1250 (gde je naša latinica) i 1251 (gde je naša ćirilica). **Nikako nemojte ostaviti ovaj korak za kasnije!** Pokušaj da se prvo pokrene skript za "uvođenje" naše podrške, pa da se tek onda instalira internacionalna podrška u *Windows 95* osnovnu "američku" verziju završava potrebom za ponovnim instaliranjem *Windows 95* operativnog sistema. Još jednom, obavezno pre prelaska na sledeći korak instalacije proverite da li postoji internacionalna podrška. Nemojte preskati pojedine faze.

Tastatura

Da bi ste mogli ćirilične i latinične TT fontove i da koristite, treba vam mogućnost promene tastatura tokom rada sa *Windows 95*. Sam *Microsoft* ponudio je dobar prebacivač standardnih tastatura. Njegova prednost je smeštenost u *Taskbar*-u i uska saradnja sa sistemom i programima pisanim za *Windows 95*. Tako, na primer, kada u *Word 7.0* promenite font sa zapadnog na *CE*, tastatura će vam se, pod *Windows 95* kada primenjujete standardni *Microsoft* prebacivač, automatski prebaciti na onu koju ste izabrali, osim USA standardne. U *Windows 95 Pan European* prebacivanje sa *CE* na *Cyrillic* TT font dovešće do toga da vam se tastatura prebaci na srpsku ćiriličnu, naravno, pod uslovom da vam je pri izboru zemlje izbor bio *Serbian*. Nedostatak je to što se ovo ne može isključiti, niti postoji mogućnost da se tokom rada sa *Word*-om (na primer) promenom tastature automatski promeni i font;osim toga, ne može se remapirati tastatura, tako da se raspored slova prilagodi pojedinom korisniku, što je kod nas često veoma potrebno usled postojanja najrazličitijih tastatura sa YU znacima (da stare navike ni ne spominjemo).

Iz tih razloga najbolje je da se prilikom prilagođavanja "američkih" *Windows 95* za jugoslovenske prilike, izbegava korišćenje *Microsoft*-ovog prebacivača tastatura i da se koriste domaći proizvodi. Odmah po izlasku *Windows 95* na tržište, izbačena su dva domaća programa te namene: *Mapper 95*, autora Dejana Jelovića iz Beograda i *EasyKey*, autora Igora Miljaševića i Milana Vidakovića iz Novog Sada. Oba se distribuiraju kao *shareware*, odnosno ako ih koristite dužni ste autorima da uplatite određenu svotu (u slučaju *Mapper*-a reč je o ekvivalentu američkih 5$, dok autori *EasyKey*-a traže 10$). Koji od njih izabrati? Pa, *Mapper 95* je 32-bitna aplikacija pisana za *Windows 95* i kao takva bi trebalo mnogo bolje da se ponaša pod *Windows 95* nego 16-bitna aplikacija, dok je *EasyKey* trenutno upotrebljiv i pod *Windows 3.1x* i pod *Windows 95*, drugim rečima on je 16-bitna aplikacija koja je inicijalno pisana za *Windows 3.1x*. *Mapper 95* nema nikakvih restrikcija u radu dok ga ne registrujete, dok je *EasyKey* više demo, ima ograničeno vreme upotrebe tokom rada.

Za trenutno aktuelnu verziju *Mapper 95 1.5* vezan je i bag koji se manifestuje zaglavljivanjem sistema ako se startuju dve kopije programa. Autor je priznao previd i najavio novu verziju, ali i trenutna je potpuno upotrebljiva ako *Mapper 95* postavite u *StartUp* folder i zaboravite na njega (osim naravno, u radu).

Prednosti oba mapera tastatura je u lakom remapiranju tastatura, kao i pravljenju novih, mogućnost importa gotovih tastatura, kao i čuvanja u fajlu napravljenih (kod *Mapper*-a). Nedostaci (mada su po nekima i to prednosti) su u nedovoljnoj sprezi sa sistemom, kao što je to, na primer, kod standardnog *Microsoft*-ovog prebacivača tastatura. Nijedan program se za sada ne može automatski instalirati, niti deinstalirati, niti se i jedan javlja u *Taskbar*-u, gde je prebacivaču tastatura prirodno mesto pod *Windows 95* operativnim sistemom. U okviru svakog od njih postoje gotove tastature za različite kodne strane koje su u upotrebi kod nas. To je nedostatak, doduše, ne autora koji su jednostavno odgovorili na navike korisnika, ali *Windows 95* bi trebalo konačno iskoristiti za prelazak na standard koji će neko i podržavati, dakle kodne strane 1250 i 1251.

Koji program izabrati? To zavisi od vas; oba programa se mogu naći na SEZAM-u, a i na mnogim drugim BBS-ovima u zemlji.

Bitmap fontovi

To su fontovi koje pod *Windows 95* možete naći u direktorijumu C:\WIN95\FONTS (naravno, ako ste *Windows 95* instalirali na disk C:\ u direktorijum WIN95) sa ekstenzijom FON. Posle instalacije *Windows 95* ima ih ukupno 18 u *Fonts* direktorijumu. Sam direktorijum nosi atribut sistemskog, svi bitmapirani fontovi imaju atribute skrivenih, a četiri i sistemskih. Tačan spisak je u donjoj tabeli:

```
8514FIX.FON    H
8514OEM.FON    H
8514SYS.FON    H
COURE.FON      H
COURF.FON      H
DOSAPP.FON     SH
MODERN.FON     H
SERIFE.FON     H
SERIFF.FON     H
SMALLE.FON     H
SMALLF.FON     H
SSERIFE.FON    H
SSERIFF.FON    H
SYMBOLE.FON    H
SYMBOLF.FON    H
VGAFIX.FON     SH
VGAOEM.FON     SH
VGASYS.FON     SH
```

Kako da dođete do naših latiničnih slova (odnosno kodne strane 1250)? Do njih ćete najlakše doći metodom (koja se čistuncima po pitanju upotrebe standardnih rešenja verovat-



no neće svideti, ali isti mogu slobodno da se žale *Microsoft*-u) zamene navedenih fontova istim tim fontovima iz prethodnih verzija *Windows*-a. *Windows 3.1 for Central and Eastern Europe* ili *Windows for Workgroups 3.11 for Central and Eastern Europe*. Pri tome nikako nemojte menjati fontove iz DOS prozora u samim *Windows 95*! Učinite to iz DOS 7.0 ili prethodne verzije DOS-a (ako ste je zadržali posle instalacije *Windows 95*).

Prilikom podizanja sistema, na poruku *Starting Windows 95*, samo pritisnite taster **F4** da biste podigli prethodnu verziju DOS-a ili **F8** da biste dobili meni iz kojeg možete izabrati DOS 7.0 ili prethodnu verziju DOS-a. U oba slučaja postupak je isti. Pozicionirate se u direktorijum C:\WIN95\FONTS. Direktorijum je, ne zaboravite, sa atributom sistemskog fajla i iz DOS-a ga komandom **dir** neće te videti. Pozicioniraćete se u njega sa **cd c:\win95\fonts**. Kada budete u njemu promenite atribute svim fontovima iz gornje tabele (preciznije, skinite im atribute sistemskih i skrivenih fajlova. Najlakše ćete to postići komandom **attrib -hs *.fon**. Posle toga možete iskopirati tih 18 fontova u poseban direktorijum za svaki slučaj.

Posle toga iz sistemskog direktorijuma starih *Windows 3.1x* (ako su vam još na disku C:\WIN31\SYSTEM) ili iz ma kojeg drugog direktorijuma, prekopirate 18 fontova sa imenima iz tabele u C:\WIN95\FONTS i postavite im potrebne atribute (ako niste imali drugih bitmapiranih fontova, bolje postavljajte atribute pojedinačno za svaki font) sa

attrib +h *.fon
attrib +s dosapp.fon
attrib +s vga*.fon

Da biste dobili naša slova u sistemskim fontovima, ostaje vam još samo da resetujete računar (ako ste u starom DOS-u) ili da otkucate EXIT (ako ste u DOS 7.0) i da podignete *Windows 95*. Sada treba još jedino da postavite *Regional Settings* na

Serbian

Osnovni problem je, ako verujete svojim očima, što takve stavke u regionalnoj podršci jednostavno nema. Gotovo da ste potpuno u pravu ... na prvi pogled. Jer podrška za nas (i ćirilična i latinična) ipak postoji i skript koji je "otključava" napisao je g. Vlada Ćalić (*vcalic@sezam.co.yu*), a do istog je došao "kopanjem" po sadržaju raznih fajlova u *Windows 95* direktorijumu. Skript možete videti dole i naziv mu je

SERBIAN.REG

```
REGEDIT4
[HKEY_LOCAL_MACHINE\System\Cur-
rentControlSet\Control\Nls\Loca-
le]
"0000081A"="Serbian (Latin)"
"00000C1A"="Serbian (Cyril)"
```

Pre nego što krenete sa instalacijom skripta obavezno proverite da li imate instaliranu *Multilanguage Support* i podršku za centralnoevropske i ćirilične zemlje. Propust da se krene u instalaciju skripta bez postojanja internacionalne podrške na koju se on oslanja, dovodi do oštećenja *Windows 95* i potrebe da se izvrši njegova ponovna instalacija. Otkucajte sadržaj skripta i snimite ga kao ASCII fajl pod nazivom serbian.reg. Zatim startujte program regedit (**Start/Run**, pa otkucajte **regedit** i kliknite na **Ok**), onda u programu regedit (*Registry Editor*) kliknite na stavku **Registry**, pa na **Import Registry File** i kada prijavi da je učitao skript kako treba, startujte **Control Panel/Regional Settings**. Trebalo bi da se pojave dve *Serbian* stavke - gornja podrazumeva ćirilični ispis, dok je donja latinična. Izabraćete, naravno, donju stavku, jer ako vam odgovara ćirilična, bolje da ste izabrali *Windows 95 Pan European* koji to nudi u startu. Podrazumeva se da je *International Support* instaliran. *Microsoft* je ipak uradio podršku za obe varijante, mada do njih nije lako doći.
Šta vam ovo donosi? Naravno, ne potpuno lokalizovan *Windows 95*, sa porukama na srpskom, ali ipak se dobijaju natpisi datuma na srpskom (ćirilicom ili latinicom), naš standardni format prikaza datuma (dan, mesec, godina), vremena (24 časa), valutu (dinar), decimalni zarez i drugo. Jednom rečju, dobijate sve ono na šta je veliki deo zapadnog sveta odavno naviknut - prilagođavanje proizvoda njima, a ne njih proizvodu.

A gde je Beograd?

Jedino što nam "kvari" uspeh u lokalizaciji *Windows 95* jeste to što u listi vremenskih zona nema Beograda. Jeste da ima mnogo drugih gradova koji su u istoj vremenskoj zoni sa nama (+1 u odnosu na Grinič), ali ipak ne živimo u Pragu ili Berlinu, niti su nam to prestonice. Srećom, tome je lako doskočiti. Donji skript (*belgrade.reg*) će u listu vremenskih zona dodati i Beograd. Prvi je skript za dodavanje nove stavke u listu vremenskih zona napisao g. Dejan Jelović (*djelovic@sezam.co.yu*) i poslao ga na SEZAM pod nazivom *yugoslavia.reg*. On, naravno, ubacuje Jugoslaviju u listu vremenskih zona. Pošto se po standardima postavljaju gradovi, a ne države ili geografska područja, skript za Beograd je možda bolje rešenje.

BELGRADE.REG



```
REGEDIT4
[HKEY_LOCAL_MACHI-
NE\SOFTWARE\Micro-
soft\Windows\CurrentVersion\Time
Zones\Belgrade]
"Display"="(GMT+01:00) Belgrade"
"Dlt"="Belgrade Daylight Time"
"MapID"="-1,66"
"Std"="Belgrade Standard Time"
"TZI"=hex:c4,ff,ff,ff,00,00,00,0
0,c4,ff,ff,ff,00,00,09,00,00,00,
05,00,02,00,00,\
00,00,00,00,00,00,00,03,00,00,00
,05,00,00,00,00,00,00,00,00,00
```

Skript otkucajte i snimite pod nazivom *belgrade.reg* u ASCII formatu. Da biste ga aktivirali startujte program *regedit* (**Start/Run**, pa otkucajte **regedit** i kliknite na **Ok**), onda u programu *regedit* (*Registry Editor*) kliknite na stavku **Registry**, pa na **Import Registry Fel**. kada prijavi da je učitao skript kako treba, startujte **Control Panel / Date/Time /Time Zone**. Trebalo bi da se pojavi stavka *Belgrade*. Izaberite nju, pa kliknite na **Ok**. I to je sve što se podešavanja vremenskih zona tiče.

A DOS?

Microsoft je svoju podršku za naša slova počeo da pruža još od verzije MS DOS 5.0. To je bilo po kodnoj strani 852. Mnogi kod nas je nikada nisu ni koristili, što je posledica šarenila raznih rasporeda koji su ovde u upotrebi. Što bi se reklo, ni manje zemlje, ni više raznih rasporeda. Doduše, sada je i tu konkurencija veća - Slovenija, Hrvatska... To je dobrim delom, na žalost, posledica inercije države. Kod mnogo većih zemalja takvih problema nije bilo. U nekadašnjem SSSR, a današnjoj Zajednici nezavisnih država, odgovarajući GOST standard donet je na vreme i uredno poštovan kao, verovali ili ne, **jedini** standard.

Kasnije je došla kodna strana 1250 koja će i ubuduće biti jedina podržana, ali i u DOS 7.0 postoji isključivo podrška za CP852. Pošto će "američki" *Windows 95* pri instalaciji sve vezano za dobijanje naših slova iz prethodnih verzija DOS-a jednostavno očistiti iz *autoexec.bat* i *config.sys* datoteka, potrebno je kasnije editovati te datoteke iz

samih *Windows 95* i dodati sledeće linije u datoteke

CONFIG SYS

```
DEVICEHIGH=C \WIN95\COM-
MAND\DISPLAY SYS CON=(EGA,,1)
COUNTRY=038,,C \WIN95\COM-
MAND\COUNTRY SYS
```

AUTOEXEC.BAT

```
C \WIN95\COMMAND\MODE CON CODE-
PAGE PREPARE=((852)
C \WIN95\COMMAND\EGA CPI)
C.\WIN95\COMMAND\MODE CON CODE-
PAGE SELECT=852
C \WIN95\COMMAND\KEYB
YU,,C \WIN95\COM-
MAND\KEYBOARD SYS
```

Ovim dobijate naša slova po kodnoj strani 852 u *Windows 95* s tim što za razliku od *Windows 95 Pan European* nema automatske konverzije iz CP852 u 1250 pri učitavanju ASCII teksta Česta greška je da se putanje odnose na stari DOS direktorijum, dok je potrebno da se odnose na direktorijum COMMAND ispod *Windows 95* direktorijuma

1250?

Da pod *Windows 95* se koristi naša latinična kodna strana 1250 i naša ćirilična kodna strana 1251 Prelazak na novi operativni sistem dobra je prilika da se konačno pređe i na standardnu kodnu stranu To je dobro ne samo zbog toga što će sama standardizacija olakšati migraciju dokumenata u računarskom obliku (ko god je pokušao recimo da dokument pisan u *Word-u* na *Windows 3 1x* sa 1250 kodnom stranom pokuša da odštampa na računaru sa američkim *Windows 3 1* sa YUSCII jem zna kakav je to problem) te će time srezati troškove već će i sva buduća podrška koju neko bude radio za nas ići isključivo za 1250

Otežavajuća okolnost mnogima je ne samo prosta navika već i činjenica da postoje verovatno čitave gomile tekstova sa našim slovima po nekom drugom rasporedu ali ovaj korak sve koji to još nisu uradili čeka pre ili kasnije Troškovi i problemi biće znatno manji ako bude pre

Ukoliko i pored svega ne želite da odstupite od vašeg rasporeda (YUSCII Računari CP852 ...) za vas je napisan program Fix All 95 (*fixall95 arj* je naziv arhive na SEZAM-u) Autori su g Nikola Krstić i g Dejan Šunderić u to vreme zaposleni u Školi računarstva Instituta "Vinča" Njega primenjujete isključivo iz DOS-a (stare verzije ili DOS-a 7 0), a nikako pod samim *Windows 95* Program radi nad GDI EXE iz direktorijuma C \WIN95\SYSTEM pri čemu se može izabrati ma koji od 12 ponuđenih rasporeda Ako izaberete 1250 moći ćete da koristite i stare *Corel* TT fontove preciznije slovo Đ Ako želite da koristite kodnu stranu 1250 i 1251 ovakvo pečovanje nije preporučljivo niti potrebno

Umesto zaključka

Lično sam uvek bio pristalica standardnih rešenja a ne prepravki ... pod uslovom da rade posao Ovo je prvi put od pojave *Windows 3 1 CEE* da sam prepravljam osnovnu verziju, a ne koristim standardno rešenje, i da čak drugima preporučujem prepravku *Windows 95* "američkih" umesto korišćenja *Windows 95 Pan European* Svestan sam činjenice da se ovakav stav neće svideti "čistuncima" po pitanju standarda ali to je njihov problem Nadajmo se da nam za *Windows 96* ovakve prepravke neće biti potrebne

Ako nemate volje ili vremena da prekucate skriptove ili se plašite greške, možete ih naći u arhivi R113W95Y ZIP na SEZAM-u ■

Autora možete kontaktirati na e-mail adresu spantic@sezam co yu

Zajedno smo jači

Iako se PowerTalk dobija uz System 7.5, retko se ko od korisnika Macintosh računara kod nas odlučio da ga instalira. Šta propuštaju, pogotovo oni koji rade u okolini sa nekoliko Macintosh korisnika, možete pročitati u tekstu koji sledi

Milan Adamov

PowerTalk se prvi put pojavio tokom 1993. godine kao deo verzije sistemskog softvera koja je nosila naziv *System 7.1 Pro*. Ustvari, *PowerTalk* je bio jedina stvar zbog koje se pomenuta verzija sistemskog softvera razlikovala od običnog *System 7.1*. *Apple* je odlučio da uz *System 7.5* standardno isporučuje *PowerTalk*, kao i mnoge druge stvari koje ranije nisu bile sastavni deo sistemskog softvera, koje su se morale posebno kupiti ili ih je proizvodio sam *Apple*.

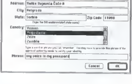
Za definiciju *PowerTalk*-a može se uzeti sledeća rečenica: *PowerTalk* pomaže ljudima da efikasnije rade zajedno, obezbeđujući mogućnost dodavanja novih kolaboracionih servisa. Oni omogućavaju nove načine komuniciranja sa drugim ljudima, očuvanje sigurnosti komunikacija, pristupanje informacijama na drugim kompjuterskim sistemima i automatizaciju rutinskih aktivnosti. *PowerTalk* sistemski softver u startu obezbeđuje neke kolaboracione servise: poštansko sanduče na desktopu, način da elektronski potpišete i odobrite dokumente, privezak s ključevima koji omogućava pristup većem broju servisa pomoću samo jednog koda za pristup, kao i mogućnost slanja fajlova drugim ljudima.

Instalacija i postavljanje

Instalacijom *System 7.5* ne instalira se i *PowerTalk*, već njega treba posebno instalirati, što važi i za *QuickDraw GX* sistemski softver. *PowerTalk* sistemski softver dolazi na dve diskete, tj. nalazi se u posebnom folderu na kompakt disku sa sistemskim softverom. Po pokretanju *Installer* aplikacije dobijamo dijalog uobičajenog izgleda koji nam nudi samo tzv. *Easy Install* opciju instalacije. Pri instalaciji *PowerTalk* sistemskog softvera ništa od njegovih komponenti ne može se izostaviti, tako da ne postoji opcija za tzv. *Custom* instalaciju. Restartovanje računara je obavezno po završetku instalacije, a po restartu se na desktopu nalaze neke nove ikone, dok su sistemski meniji dobili nekoliko novih stavki.

Jedna od ikona je pominjano poštansko sanduče, a reč *Mailbox* i stoji ispod ikone koja predstavlja sanduče. Većina korisnika odmah po instalaciji, privučena ovim natpisom, prvo klikne na ovu ikonu i odmah se susretne sa drugim ugrađenim kolaboracionim servisom, bez kojeg se drugim *PowerTalk* kolaboracionim servisima ne može ni prići. Reč je o, takođe već pomenutom, privesku s ključevima (*Key Chain*), a prvo što korisnik mora da uradi jeste da odredi ono što *PowerTalk* u originalu naziva *Key Chain Access Code*, odnosno jednostavnije rečeno, lozinku kojom će otvarati svoj privezak da bi pristupio ključevima na njemu. Kada je korisnik postavio svoju lozinku za *Key Chain*, natpis *Mailbox* ispod ikone poštanskog sandučeta na desktopu se menja u ime koje je korisnik uneo prilikom određivanja lozinke za otključavanje priveska sa ključevima, a to ime može biti različito od onog koje stoji kao ime vlasnika računara u *Sharing Setup* kontrolnom panelu.

Ne pominjem slučajno ovaj kontrolni panel, jer nas *PowerTalk* po instalaciji odmah upozorava da proverimo da li smo pravilno postavili njegove parametre, posebno ime računara, jer ono aktivno učestvuje kao parametar u adresiranju elektronske pošte korišćenjem usluga *PowerTalk* sistemskog softvera. Takođe veoma je važno da su ispravno postavljeni parametri koji se postavljaju pomoću *Date & Time* kontrolnog panela, a to su vremenska zona, vreme kao i podatak o letnjem, odnosno zimskom računanju vremena. Vremensku zonu postavite pronalaženjem linije sa tekstom *Belgrade, Yugoslavia* kada kliknete na odgovarajuće dugme u *Date & Time* kontrolnom panelu, čime ćete dobiti tekst "*Belgrade is a city in the current time zone*", a *check box* ispod toga, pored kojeg stoji tekst "*Daylight Savings Time*", treba da bude isključen kada važi zimsko računanje vremena, kao što je sada slučaj.



Za isključivanje *PowerTalk* servisa na duže vreme, kao i za podešavanje još nekih parametara vezanih prvenstveno za *Key Chain*, služi *PowerTalk Setup* kontrolni panel. Inače, iako se preporučuje da računar na kojem je instaliran *PowerTalk* ima 16 Mb RAM-a, on će sasvim lepo raditi i na računaru sa 8 Mb RAM-a, pod uslovom da *System* nije preterano obogaćen raznim ekstenzijama koje oduzimaju dragocenu memoriju.

Privezak za ključeve

Key Chain Access Code služi nam za pristup svim "ključevima" koje ćemo staviti na privezak, a pod ključevima se podrazumevaju lozinke za pristup fajl serverima na *AppleTalk* mreži, kao i lozinke za pristup ostalim kolaboracionim servisima koji se mogu ugraditi u *PowerTalk*. Na primer, kada prvi put po instalaciji *PowerTalk*-a i otključavanju *Key Chain*-a pristupimo nekom *AppleShare* fajl serveru, *PowerTalk* će nas upitati da li želimo da nam postavi "ključ" za pristup fajl serveru na *Key Chain*. Pod ključem se podrazumeva sve što je potrebno obaviti da bi se pristupilo na fajl server (odlazak u *Chooser*, biranje servera, unošenje lozinke). Kada kreira ključ koji stavlja na privezak, *Power Talk* praktično čini isto što i korisnik kada kreira *alias* diska fajl servera (štedi broj koraka potrebnih da bi se pristupilo serveru).

Isto pitanje *PowerTalk* će postavljati za svaki server na koji se korisnik bude prijavljivao po njegovoj instalaciji ukoliko nije isključen. Elem, kada se otključa *Key Chain* unošenjem *Key Chain Access* koda, a to se može učiniti pokušajem pristupanja bilo kojem instaliranom *PowerTalk* servisu ili jednostavnim pozivanjem opcije *Unlock Key Chain* iz **Special** menija u *Finder*-u, nije više potrebno unositi lozinku za pristup serveru, niti jednu drugu lozinku čiji se ključ nalazi na privesku.

Da bi se osigurali podaci prilikom ostavljanja uključenog računara na neko vreme,

dovoljno je zaključati *Key Chain* takođe pomoću **Special** menija i odjaviti diskove servera sa desktopa i vaši podaci su sakriveni *Key Chain Access* kodom. S druge strane, eventualni posetilac može klikom na *Cancel* dugme u dijalogu za unošenje lozinke za pristup da dobije uobičajeni dijalog za prijavljivanje na fajl server i prijavi se pod svojim imenom i lozinkom sa vašeg računara. Pomoću *PowerTalk Setup* kontrolnog panela moguće je postaviti da se zaključavanje *Key Chain*-a obavi automatski posle određenog perioda neaktivnosti, što je zgodno za zaboravne korisnike. Takođe je preko pomenutog kontrolnog panela moguće postaviti da se dijalog sa zahtevom za unošenjem *Key Chain Access* koda pojavi odmah po podizanju sistema, čime se omogućava da *PowerTalk* odmah obavesti korisnika o elektronskoj pošti pristigloj u međuvremenu.

Inače, preko *Key Chain*-a se novoinstalirani kolaboracioni servisi puštaju u rad, izbacuju se ili se postavljaju njihovi parametri potrebni za njihov rad. Po instalaciji *PowerTalk*-a *Key Chain* je prazan, nema nijednog kolaboracionog servisa, iako su dva standardno ugrađena u *PowerTalk*: *Direct AppleTalk mail* i *PowerShare*. Oba predstavljaju sistemski podržanu elektronsku poštu, s tim što se u drugom slučaju mora posedovati *PowerShare* server softver instaliran na nekom od računara, a *PowerShare* se posebno kupuje.

Logično je da ćete prvo isprobati ugrađene mogućnosti koje pruža *Direct AppleTalk mail* servis. Razlika je u tome, pored još nekih detalja, što je kod elektronske pošte bez *PowerShare* servera moguće da jedna poruka nikada ne stigne na odredište. Naime, da bi pomoću *Direct AppleTalk mail* servisa poslata poruka stigla na odredište, računar s kojeg je poslata i onaj na kojem se prima moraju bar neko vreme da budu uključeni istovremeno. Na kompakt disku sa *System 7.5* nalazi se i nekoliko dodatnih kolaboracionih servisa, poput *Direct Dialup* servisa i nekoliko servisa koji su demonstracioni, poput recimo nekoliko servisa za razmenu elektronske pošte preko Interneta ili



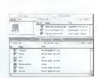
Compuserve mreže, kao i servis za slanje faksova korišćenjem *PowerTalk*-a. Koji će od servisa koje smo postavili u *Key Chain* biti aktivan određujemo pomoću *I'm at* opcije iz **Special** menija, gde možemo uključiti ili isključiti određene servise u zavisnosti od toga da li smo na poslu, kod kuće ili na putu. Na primer, podesićemo da nam je aktivan *Direct Dialup* ako idemo negde na put da bi sa *PowerBook* prenosnog računara, na kojem smo takođe uključili *Direct Dialup*, modemom mogli da proverimo elektronsku poštu pristiglu u međuvremenu na računar u kancelariji.

Katalozi

Katalozi su skupovi informacija o ljudima i stvarima s kojima možete da radite. Na primer, katalog može da sadrži informacije o korisnicima kompjutera, grupama korisnika i uređajima na mreži računara. Informacija koja je sadržana u katalogu zavisi od vrste kataloga. Instalacijom *PowerTalk*-a dobijate dva: *AppleTalk Catalog* i *Personal Catalog*, s tim što kasnije možete dodavati koliko želite ličnih (*Personal*) kataloga, dok *AppleTalk* katalog jednostavno predstavlja spisak svih tog trenutka uključenih računara na vašoj *AppleTalk* mreži sa uključenim *PowerTalk* softverom i svih računara koji imaju uključen *file sharing*, odnosno predstavljaju *AppleShare* fajl servere. Novi *PowerTalk* kolaboracioni servisi koje ćete eventualno kasnije instalirati mogu kreirati svoje kataloge.

Lični katalog (*Personal Catalog*) možete iskoristiti da sebi olakšate komunikaciju sa drugim ljudima, ubacivši u njega sve adrese koje ste našli u *AppleTalk* katalogu. *Direct AppleTalk mail* adrese su u formi *<ime računara>@<AppleTalk zona>*, ali im možete dati smisleniji naziv tako što ćete kreirati *alias* adrese iz *AppleTalk* kataloga (u kojem vam nisu dozvoljene izmene) u lični katalog, a potom tu stavku preimenovati na isti način kao što se u *Finder*-u preimenuju fajlovi i folderi. *Alias* adrese iz *AppleTalk* kataloga kreira se jednostavnim prevlačenjem stavke iz *AppleTalk* kataloga preko ikone ili na otvoren prozor ličnog kataloga.

Sledeća stvar koja se može uraditi u cilju efikasnijeg korišćenja *PowerTalk*-a je kreiranje informacionih kartica sa podacima o ljudima s kojima razmenjujete elektronsku poštu, kao i kreiranje grupa sa *Direct AppleTalk* adresama i informacionim karticama, što može, recimo, da omogući jednostavnije slanje cirkularnih poruka putem elektronske pošte. Informaciona kartica može da sadrži sve relevantne podatke o jednoj osobi, od uobičajenih podataka o imenu, prezimenu, nadimku, kućnoj adresi i slično, do svih adresa za elektronsku poštu koje ta osoba poseduje.



Applemail i mailer

Mailer je deo *PowerTalk* softvera kojim se adresira elektronska pošta, šalje korespondencija iz bilo koje aplikacije, pod uslovom da su programeri pisali aplikaciju tako da uopšte radi sa *PowerTalk*-om. Ukoliko su predvideli, korisnik je u stanju da pošalje dokument na kojem radi pomoću aplikacije direktno iz nje, bez korišćenja nekih drugih programa za slanje elektronske pošte. Na žalost, takve aplikacije su još prilično retke, tako da smo primorani da poruke šaljemo korišćenjem *AppleMail* programa koji se dobija uz *PowerTalk*. Inače, *AppleMail* upravo koristi *Mailer* da bi korisnik adresirao i slao poruke.

Kada se šalje neka poruka, *Mailer* automatski u polje za pošiljaoca poruke postavi ime koje smo odredili unoseći svoj *Key Chain Access* kod. Polje za temu (*Subject*) mora da se popuni (ukoliko baš želite da bude prazno, unesite jedan razmak), kao što mora da se navede i bar jedan primalac poruke. Klikom na dugme pored kojeg piše *Recipients* otvara nam se malo prozorče iz kojeg biramo primaoca ili primaoce poruke. S leve strane imamo niz dugmića od kojih nam prvo, sa izgledom otvorene knjižice, prikazuje sve stavke koje imamo u svom ličnom katalogu, a ono ispod njega, sa ikonicom zemaljske kugle, lista *AppleTalk* katalog i prikazuje trenutno pristupačne *PowerTalk* adrese na *AppleTalk* mreži. Dugme sa nacrtanom lupom služi za pretraživanje kataloga u potrazi za određenim tekstom, dok poslednje dugme u nizu, sa nacrtanom olovkom, služi za direktno unošenje adrese primaoca. Ispod spiska se nalaze dugmići kojima određujemo da li šaljemo poruku direktno primaocu, kao kopiju ili kao tzv. slepu kopiju (*blind copy*). Ukoliko se ne navede da primalac prima slepu kopiju, svi ostali primaoci poruke videće da je poruka poslata i njemu.

Ukoliko želite da prikačite neki fajl ili folder uz poruku koju šaljete, to možete uraditi na dva načina. Prvi je da se klikne na dugme *Enclosures* u okviru *Mailer*-a i da se kroz standardni fajl selektor dijalog odabere željeni fajl ili folder, a za svaki sledeći fajl ili folder

ko, se želi prikačiti postupak treba ponoviti. Broj fajlova i foldera koji se mogu prikačiti uz poruku praktično je neograničen, a fajlovi se bukvalno kopiraju u poruku. Drugi način je jednostavno prevlačenje fajla iz prozora foldera u kojem se nalazi u pravougaonik u kojem se, u okviru *Mailer*-a, izlistavaju prikačeni fajlovi i folderi. Na sličan način se, dakle pomoću *drag and drop*, mogu i slati fajlovi direktno iz *Finder*-a tako što se fajl prevuče preko informacione kartice primaoca ili njegove *Direct AppleTalk* mail adrese.

AppleMail omogućava da se u poruku ubaci praktično sve što se i inače može iskopirati korišćenjem sistemskog klipborda: tekst, grafika, zvuk i *QuickTime* film. Poruku iz njega možemo poslati u dva oblika, pojedinačno ili istovremeno. Prvi je oblik koji možemo nazvati *AppleMail* porukom, kada se poruka primaocu šalje u *AppleMail* formatu, što će reći da primalac mora da ima *AppleMail* aplikaciju da bi mogao da je pročita. Dok je bude čitao, moći će da pregleda sve što ta poruka sadrži, tačnije moći će da pogleda *QuickTime* film koji smo eventualno ubacili u poruku, čuje neki efektan zvuk ili naš glas itd. Drugi oblik naziva se *snapshot* i time se šalje samo slika poruke, tako da primalac ne mora da ima *AppleMail* aplikaciju ili nije u stanju da manipuliše zvukovima, slikama, čak ni tekstom poruke, jer je cela poruka poslata kao slika.

Poštansko sanduče

Poruke sa svih servisa dolaze u jedno jedinstveno poštansko sanduče za sve njih. Ovo poštansko sanduče sastoji se iz dva dela i oni se nazivaju *In Tray* i *Out Tray*. U *In Tray* se smeštaju sve pristigle poruke i on se otvara kada otvorimo poštansko sanduče, a poruku čitamo dvostrukim klikom na nju, čime se ona učitava u *AppleMail* ili neku drugu aplikaciju koja ume da pročita takav format poruke. Da bismo otvorili *Out Tray* koristimo opciju menija Mailbox koji se pojavljuje desno od Special menija u *Finder*-u kada otvorimo poštansko sanduče, a *Out Tray* sadrži



poruke koje smo mi slali drugim ljudima.

Mailbox meni ćemo koristiti za postavljanje preferenci za poštansko sanduče, odnosno za određivanje načina na koji želimo da budemo obavešteni o pristizanju nove poruke, posle koliko dana da sistem automatski obriše poslate poruke iz *Out Tray*-a. Opcija kojom dozvoljavamo da gosti kreiraju svoje poštanske sandučiće funkcioniše samo ako imamo *PowerShare* server.

Ukoliko dođe do nekog problema u slanju ili prijemu poruka, *PowerTalk* će nas obavestiti iscrtavanjem trougla sa znakom uzvika u prozoru za *In Tray*, a klikom na trougao dobiće se detaljnije objašnjenje problema. Na primer, jedan od češćih problema je da neka cirkularna poruka dugo stoji u *Out Tray*-u, jer se nije mogla, iz ko zna kojeg razloga, poslati nekom korisniku. Status svake od poruka koju smo poslali možemo proveriti otvaranjem *Out Tray* prozora, a on može biti jedan od tri: *sending*, *done* ili *waiting*. Prvi (*sending*) znači da *PowerTalk* tog trenutka šalje poruku nekom od primalaca, a drugi (*done*) znači da je poruka uspešno poslata svim primaocima i da se čeka da prođe period koji smo odredili u preferencama poštanskog sandučeta da bi bila automatski obrisana. Naravno, uspešno poslate poruke možete odmah i sami obrisati povlačenjem u kantu za đubre, a automatsko brisanje poslatih poruka posle određenog vremena služi da vam poruke koje šaljete ne oduzimaju prostor na disku u nedogled ako zaboravljate da ih sami obrišete. Treći status poruke (*waiting*) znači da *PowerTalk* čeka da isporuči poruku, a to može da bude iz više razloga. Možda je reč o cirkularnoj poruci na mreži koja nema *PowerShare* server, pa *PowerTalk* čeka i proverava da li se *Macintosh* primaoca pojavio na mreži, ili je možda reč o poruci poslatoj korišćenjem *Direct Dialup* servisa koji se može podesiti da poštu šalje u tačno određeno vreme da bi se, recimo, iskoristila jeftinija tarifa za PTT usluge.

Digitalni potpis

PowerTalk obezbeđuje način za proveru verodostojnosti primljene poruke ili fajla korišćenjem aplikacije koju instalira na disk. Reč je o aplikaciji *DigiSign Utility* koja služi za elektronsko potpisivanje dokumenata. Na žalost, ovu veoma lepu mogućnost teško ćemo imati prilike da u dogledno vreme iskoristimo kod nas.

Naime, da biste mogli da elektronski potpisujete fajlove i poruke, morate da dobijete svoj "potpis" (*signer*), a da biste to dobili morate da obavite nekoliko koraka. Prvo morate da kreirate zahtev za izdavanje pot-



pisa korišćenjem *DigiSign Utility* aplikacije koja će kreirati dva fajla: formular zahteva i neodobreni *Signer* fajl sa enkodiranim informacijama koje će kasnije postati odobreni fajl za potpisivanje. Zahtev se šalje agenciji koja ima odobrenje za notiranje dozvola za *Signer* fajl, što je u našem slučaju još nemoguće, mada Jugoslavija postoji u spisku zemalja prilikom popunjavanja zahteva za *Signer* fajl. Notirani formular šalje se firmi *RSA Data Security Inc.*, a ona po odobravanju šalje fajl kojim ćemo neodobreni *Signer* fajl pretvoriti u odobreni i potpuno funkcionalni *Signer* fajl.

Sve u svemu, kada se otvori prozor sa informacijama za potpisan fajl, uz uobičajene podatke s desne strane prostora za komentar stoji nova ikona. Klikom na nju pokreće se proces verifikacije potpisa, po čijem se uspešnom završetku ispisuju podaci o osobi koja je fajl digitalno potpisala. Svaki *PowerTalk* korisnik, čak i onaj bez sopstvenog *Signer* fajla, može da verifikuje potpisan dokument.

Iz ličnog iskustva

Moram priznati da sam bio skeptičan po pitanju instaliranja *PowerTalk*-a na računare u firmi u kojoj radim, pogotovo što sam imao loše iskustvo sa *PowerTalk*-om koji je dolazio uz *System 7.1 Pro*. Naime, tada je to bila stvar koja je previše opterećivala resurse mašine.

Sada, uz verziju *PowerTalk*-a koja stiže uz *System 7.5*, to više nije slučaj. Prošlo je skoro godinu dana od kada sam ga instalirao na svaki od računara u mreži u kojoj su i dva, za današnje pojmove, "slabića" od računara kao što je *Macintosh* LC III sa po 8 Mb RAM-a. *PowerTalk*, a i *System 7.5*, na njima rade veoma lepo uz uobičajene radne zadatke koje ove mašine obavljaju. Da ne govorim o tome kako bih izazvao revolt korisnika kada bih, kojim slučajem, sada uklonio *PowerTalk* sa njihovih računara.

Dakle, zaključak je da *PowerTalk* vredi instalirati, pogotovo ako imate umrežene Mekove u raznim prostorijama. Mnogo je lepše otkucati i poslati poruku nego se dovikivati po prostorijama ili bezuspešno telefonom pokušavati da se dobije zauzet lokal.

Autora možete kontaktirati na madomov@sezam.co.yu i madomov@osmeh.fon.bg.ac.yu n


GODINAMA BEZ PRAVE KONKURENCIJE
IMTEL
Mini 4.208
486DX2-66 MHz VLB
Standard 5.584
486DX4-100 MHz VLB
COLOR 14"
Professional 6.736
486DX4-100 MHz PCI
FLOPPY DRIVE 1 44MB
COLOR 14"
De luxe 9.832
586-90 MHz PCI
COLOR 14"
Microsoft Windows 95
DINAASKE CENE
KURSEVI
MREŽE
POSLOVNI
SOFTVER
Oni koji koriste naše računare znaju zašto je to tako
IMTEL computers
TEL: (011) 134-516, 222-40-78, 135-602, 135-420, (021) 58-344 FAX: (011) 142-164
INTERNET: prodaja@imtel co.yu
IMTEL BBS: 141-992
BULEVAR LENJINA 165-b, N.BEOGRAD
BROJ 17 U ČITALAČKOM SERVISU


Bul . Oslobodenja 59
Novi Sad tel. 623-928
ASYS
c o m p u t e r s
Matrični i ink-jet štampači
najpoznatije svetske firme
EPSON
Računarska oprema ,
softver i servis
PRAVO DRUŠTVO ZA VAŠ KOMPJUTER

Totalna kontrola

In[illegible]ski kontroleri [illegible] ko[illegible] trendo[illegible] ještavaj[illegible] znolikost [illegible] području [illegible] umreti. Ostaje [illegible] vidimo kako će [illegible] to izgledati.

Industrijski kontroleri predstavljaju onaj deo kompjuterske industrije koji je nevidljiv za običnog korisnika. O njima se (uglavnom) ne piše u časopisima, na kompjuterskim sajmovima se pojavljuju (ako se uopšte pojavljuju) u izdvojenim halama i veoma mali broj korisnika računara uopšte zna za njih. Međutim, to ne znači da se u ovoj oblasti dešavaju išta manje zanimljive stvari od onih koje već nekoliko godina drmaju industriju PC računara. Šta više, događaji se razvijaju tako da se sudbina industrijskih kontrolera i programabilnih logičkih kontrolera sve više vezuje za PC arhitekturu.

Istorijat

Industrijski kontroleri počeli su da se pojavljuju krajem sedamdesetih i početkom osamdesetih godina. Tada oni nisu bili specijalizovane mašine već su to bili računari prilagođeni za jednostavne operacije obrade ulaznih signala i korišćeni su uglavnom u laboratorijskim uslovima za kontrolu i obradu merne opreme. Kako je računarska tehnologija napredovala, napredovali su i kontroleri, ali su pravi procvat doživeli tek sa pojavom mikroprocesora.

Kada je pojavom mikroprocesora postalo moguće da se ceo računar spakuje u relativno mali prostor i kada je postalo moguće da računari rade izvan prostorija sa održavanom mikroklimom, pala je poslednja prepreka koja je sprečavala njihovu upotrebu u industriji. Do tada inženjerima je već postalo jasno da je za malo komplikovanije zadatke automatske kontrole i upravljanja mnogo lakše koristiti računar nego projektovati logičku šemu. Takođe, za zadatke koji su zahtevali memoriju ili aritmetičke proračune digitalna logika nije bila upotrebljiva, zato su se mikroprocesori nametnuli kao logičan izbor. Imali su dovoljno dobre ulazno-izlazne mogućnosti da bi se mogli prilagoditi potrebama upravljanja, a 16-bitni adresni prostor koji su mikroprocesori dobili vrlo skoro po svom nastanku obezbeđivao je dovoljno memorije i za veoma komplikovane zadatke kontrole i upravljanja.



Tada je polako počelo bujanje različitih kontrolera, specijalizovanih za određene namene. U početku su velike firme počele da ih proizvode i ugrađuju kao upravljačke jedinice u sopstvenu opremu. Međutim, posle samo nekoliko godina postalo je jasno da postoji ogromno tržište (naročito u oblasti mašinske industrije) koje bi moglo da koristi industrijske kontrolere, ali nije bilo u stanju da plati visoke troškove razvoja specijalizovanih kontrolera. Zato su uskoro počeli da se javljaju univerzalni kontroleri koji su imali univerzalne sposobnosti. Ovakve kontrolere kupovale su mnogobrojne firme koje su ih ugrađivale u svoju opremu. Pisanje programa je takođe prepušteno programerima kupca. To je bilo sasvim logično, jer su oni mnogo bolje poznavali osobine procesa koji se kontroliše.

Ovo nas je dovelo do sledeće etape u razvoju kontrolera, a to je pojava programa koji su pravljeni da olakšaju programiranje kontrolera. Oni su postali neophodni čim su se industrijski kontroleri toliko raširili da više nije bilo dovoljno programera upoznatih sa problematikom procesa. Zbog toga je i običan inženjer (mašinski ili elektro) koji je posedovao malo veće znanja iz programiranja i automatike, a učestvovao je u razvoju mašine ili uređaja u koji se ugrađivao kontroler, mogao da vrši prilagođavanja programa još u periodu razvoja.

Pravci razvoja

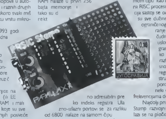
Zlatno doba industrijskih kontrolera trajalo je tokom celih osamdesetih godina i u tom periodu su se mnogi proizvođači kompjutera okušali na ovom polju. Ali, kao i kod svake nove tehnologije, pionirska vremena kad-tad prođu i dođe vreme da se stvari srede. Prvi korak u sređivanju stvari je ustanovljavanje standarda.

S obzirom na to da su brojne velike firme već formirale sopstvene standarde u ovoj oblasti, standardizacija je dugo stajala u mestu. Postignuti su dogovori oko komunikacionih standarda, konektora i nekih drugih sitnijih detalja, ali je pitanje hardverske platforme ostalo nedirnuto. Međutim, postalo je očigledno da dom[illegible]

Mikrokontroler u svakom uređaju

Dok se u burnom moru industrijskih kontrolera vodi bitka u kojoj su glavna oružja frekvencije procesora i širine magistrala, u "sigurnoj luci" mikrokontrolera se u tišini razvoj odvija u sasvim drugom pravcu. Tu se cene male dimenzije, niska potrošnja i kompaktan dizajn. Ovaj novi pravac u razvoju mikrokontrolera je čak dobio i svoju šaljivu skraćenicu *IBP* (*itty bitty processor* – majušni procesor). Do ove "maksiminijaturizacije" došlo je zbog potpuno nove namene za koju se mikrokontroleri danas koriste. To je kontrola kućnih aparata, uređaja i sklopova u automobilima i drugim vozilima i raznih drugih uređaja i sprava (tako npr. skoro svaki miš ili *trackball* u sebi sadrži neku vrstu mikrokontrolera).

Istraživanja koja je još 1993. godine sprovela "*Motorola*", jedna od vodećih firmi na ovom području, pokazuju da će do 2000-te godine prosečno domaćinstvo u svojim aparatima imati preko 200 mikrokontrolera ugrađenih u televizore, videorekordere, telefone, muzičke uređaje itd.



Tipičan *IBP* čip današnjice na sebi ima kompletan ROM (ili EEPROM ili FLASH RAM) i RAM i mali broj nožica (od 16 do 20) koje su sve osim nekoliko najneophodnijih posvećene ulazima i izlazima. Osim toga, za razliku od "velikih" procesora koji su tek nedavno prešli na rad sa naponom od 3V, mikrokontroleri već nekoliko godina rade na 2V (a i na manjim naponima). Srce ovakvih *IBP*-a je ozbiljan osmobitni mikroprocesor sa akumulatorima, indeks registrima, stekovima i dobro koncipiranim setovima instrukcija. Svi imaju paralelne izlazno-ulazne portove, skoro svi imaju jedan ili više tajmera, a mnogi imaju ugrađen serijski ulaz-izlaz, dok neki čak imaju mogućnost povezivanja sa analognim konvertorima. Jednostavno govoreći, njihova namena je da obezbede "inteligenciju" u krajnje specijalizovanim primenama. Jedan *IBP* čip može da zameni veliku količinu "neinteligentne" digitalne elektronike. I naravno, ako se zadatak koji treba da obavlja *IBP* malo promeni, reprogramirate ga i problem je rešen.

Tri firme koje vode glavnu reč na ovom polju su *Motorola*, *Zilog* i *Microchip*.

Motorolina serija mikrokontrolera 68HC05 vuče poreklo iz čuvenog Motorolinog mikroprocesora 6800. U poređenju sa svojim pretkom, HC05 serija je, zbog svoje namene, i proširena i sužena u mogućnostima. Na primer, 20-pinski član ove porodice sa oznakom 68HC05J1A ima jedan 8-bitni akumulator i jedan 8-bitni *index* registar, za razliku od 6800 koji je imao dva 8-bitna akumulatora i 16-bitni indeks registar. Ovakva koncepcija je sasvim razumljiva: 68HC05J1A ima samo 64 bajta korisničkog RAM-a (i 1240 bajta ROM-a) te je adresni prostor od 64K sasvim nepotreban. Pri tome se svi ulazno-izlazni portovi i korisnički RAM nalaze u prvih 256 bajta memorije i tako su direktno adresabilni preko indeks registra. Ulazno-izlazni portovi se, za razliku od 6800, nalaze na samom čipu.

Motorola nastavlja da smanjuje dimenzije i potrošnju svojih mikrokontrolera. Tako je u februaru je predstavljena serija HC05 procesora koji rade na 1.8V. Jedan od njih, 68HCL05K0, smešten u šesnaestopinskom kućištu, uz sve ostale funkcije ima i 512 bajta ROM-a i 32 bajta RAM-a.

Zilog je razvio seriju mikrokontrolera Z86xx kojima je svima zajedničko "srce", mikrokontroler Z8, oko kojega su napravljene razne konfiguracije. Postoje dve verzije ovih mikrokontrolera: Z86Cxx, koji rade na 3V, i Z86Lxx, koji rade na 2V. Pri tome i jedni i drugi u jeftinijim verzijama mogu da rade na frekvencijama do 12MHz, a u skupljim do 20 MHz. Z86xx familija nema ni akumulator ni *index* registar. Umesto toga, cela korisnička RAM memorija je raspodeljena u registre kojima se može pristupati kao što se pristupa akumulatorima i indeks registru. Korisnički ROM je velik 1KB, dok su ulazno-izlazne linije raspoređene u tri porta (portovi 1 i 3 po 3 linije, a port 2 – 8 linija). U zavisnosti od modela, Z86xx mikrokontroleri od dodatne "opreme" imaju *watchdog* tajmere, serijske interfejse i čak analogne komparatore.

Microchip-ova serija PIC16C5x mikrokontrolera je podvrsta većeg procesora PIC16x. Na sebi imaju ulazno-izlazne pinove, ali nemaju ugrađen korisnički ROM, već imaju pinove za direktnu vezu sa serijskim EEPROM čipom. Korisnički RAM je ograničen na 25 osmobitnih memorijskih lokacija koji su kao i kod *Zilog*-ovih mikrokontrolera pretvorene u registre. Od ostalih interesantnih karakteristika treba spomenuti mogućnost postojanja A/D konvertora na samom čipu, kao i set instrukcija koji podseća na RISC procesore. Celokupan set instrukcija sastoji se od samo 33 instrukcije koje su sve dužine po 12 bita. Zbog ovako ograničenog broja instrukcija programiranje može izgledati pomalo čudno. Postoji samo jedna bezuslovna GOT naredba, a sva grananja obavljaju se sa nekoliko instrukcija koje (sve uz razlike) rade tako što u zavisnosti od sadržaja nekog od registara preskaču sledeću instrukciju. Svi članovi PIC16C5x porodice imaju sleep režim, a neki od njih mogu da rade na frekvencijama do 20 MHz.

Najbolji primer primene *IBP*-a je *Basic Stamp* razvojni sistem firme *Parallax*. Nalazi se na pločici dimenzija 38x64 milimetra i sastoji se od dva čipa (PIC16C56 i 256-bajtni serijski EEPROM), oscilatora učestalosti 4 MHz, kontrolera napona, nekoliko pasivnih elektronskih komponenti i priključka za 9 voltnu bateriju. Programira se nešto osakaćenom tokenizovanom verzijom *BASIC*-a koja podržava rad sa ulazno-izlaznim portovima. U EEPROM staje oko 80 linija tokenizovanog *BASIC* koda, što je sasvim dovoljno za većinu upotreba. Uz ovaj kontrolerčić dobijate i sav softver potreban za razvoj programa i njegovo prebacivanje preko serijske veze iz računara u EEPROM.

Basic Stamp je u svakom pogledu izuzetno inženjersko dostignuće, i hardversko i softversko. S obzirom na to da se trajno programiranje vrši svaki put kada prebacite program iz računara u EEPROM kontrolera, ovaj razvojni sistem može vrlo ozbiljno da se iskoristi i kao konačna verzija kontrolera: treba samo da povežete ulaze i izlaze na njegovu matricu i da spojite napajanje. *Basic Stamp* je urađen dovoljno solidno i kompaktno da se može bez prilagođavanja ugraditi u bilo koji uređaj.

nantna računarska platforma. *Intel*-ova serija 80x86 procesora ne može biti zaustavljena u prodoru na ovo tržište iako još uvek razni proizvođači odolevaju ovom pritisku i proizvode uređaje bazirane na nekim drugim hardverskim rešenjima.

Jednostavno, u *status quo* koji vlada po pitanju standardizacije već duže vreme ubacuju se *IBM* i druge kompanije proizvodeći industrijske kontrolere koji su u stvari PC računari prilagođeni za rad u industrijskim uslovima. Ovakvi industrijski kontroleri izbacuju konkurenciju prvenstveno niskom cenom i kompatibilnošću sa PC računarima i polako se nameću kao standard. Oni koriste ulazne i izlazne kartice koje se na matičnu ploču povezuju preko ISA magistrale standardnim edge konektorima. Kao operativni sistem koristi se *DOS*, *Windows* ili *QNX*.

Budući da prosečni programi za kontrolu procesa imaju oko 64 Kb, postavlja se pitanje zašto je za standard odabrana ovako jaka platforma. Kao prvo, tu je nastup sa filozofijom "grube sile" koju zastupaju veliki proizvođači (*Intel*, *IBM*...). To što industrijski kontroler koji radi pod *Windows*-ima, ima, recimo, 8 MB memorije i disk od 250 MB, a vaša aplikacija može da se smesti u 64Kb, nikoga ne zabrinjava. Jer danas 8MB memorije košta otprilike isto koliko i 64 kilobajtni EEPROM čip pre 6-7 godina, a cena PC platforme je, zbog masovne proizvodnje, niža od bilo koje specijalizovane varijante.

Drugi razlog je u tome što se ovakvom koncepcijom omogućava da se sa jednog mesta vrši potpuna kontrola sistema. Do pojave PC kontrolera ste, ako ste želeli da imate kvalitetan vizuelni prikaz dešavanja u sistemu kontrole, morali da imate zaseban PC računar sa zasebnim programom, koji bi komunicirao sa kontrolerom i preuzete podatke prikazivao na ekranu. Ovaj sistem je, osim visoke cene, imao i malu brzinu, jer su sve informacije između računara i kontrolera obično išle relativno sporim serijskim vezama, tako da je količina podataka koja je mogla biti prenesena na ovaj način bila mala.

Promena u hardveru kontrolera danas donosi i neke promene u filozofiji kontrole. Zbog niske cene osnovnog sistema, ulazno-izlaznih blokova, kao i zbog velike brzine procesora, tendencija je da se kompletna kontrola svih veličina kontrolisanog sistema rešava softverski. Čak i neke stvari koje su se ranije rešavale lokalno (bez uticaja kontrolera) danas idu kroz kontroler. Na taj način je moguće ostvariti kompletan uvid u stanje sistema koji se kontroliše. Pri tome se svi podaci prikazuju na ekranu i potpuno su pregledni. Ako se radi pod Windows-ima, još je lakše organizovati vizuelne podatke, jer postoji mogućnost da se prikaz podeli na nekoliko ekrana.

Programiranje

Zbog poznatih problema vezanih za programiranje u mašinskom jeziku, bilo je potrebno naći metod za njihovo jednostavno pro-

Kako se to radi kod nas

Za razliku od tržišta PC računara, na kojem je nemoguće konkurisati masovnoj proizvodnji sa Tajvana ili iz Kine, industrijski kontroleri zbog svojih malih serija omogućavaju domaćim firmama da ih same proizvode. Pri tome pod proizvodnjom ne podrazumevam samo sklapanje gotovih komponenti (što se kod nas obično naziva "proizvodnja"), već ozbiljnu maloserijsku proizvodnju sa sopstvenim razvojem.

Firma koja kod nas prednjači u primeni PC standarda kao osnove za svoje proizvode je *Electronic Design* iz Beograda. Od samog početka svog postojanja ljudi iz ove firme su se opredelili za razvoj baziran na PC platformi, još od 1985, kada su prvi u svetu napravili PC kompjuter na jednoj kartici Evropa formata. Ideja vodilja bila je da se napravi potpuno PC kompatibilan industrijski kontroler. Ovaj pristup se potvrdio kao ispravan, jer dešavanja u svetu potvrđuju da PC standard počinje da preovladuje na tržištu. Na ovaj način oni svetsku dominaciju PC kontrolera dočekuju spremno, sa tri modela kontrolera i potpunom gamom ulazno-izlaznih kartica. U njihovom proizvodnom programu nalaze se modeli *R2D2*, *C3PO* i *C3PO II*.

R2D2 je bazni model industrijskog računara baziran na PC arhitekturi i 386, 486 ili Pentium procesoru. Na matičnu ploču može da se stavi sedam kartica u dvostrukom Euro formatu. Zbog standarda koji su uspostavili veliki svetski proizvođači (*IBM*, *Siemens*, *Intel*) priključivanje ploča se standardno vrši edge konektorima, mada je na zahtev kupca moguće ugraditi i druge konektore. Ovaj i ostali kontroleri koriste ISA magistralu za vezu sa karticama.

Svi kontroleri imaju mogućnost da se u njih ugradi klasičan IDE disk, disketa, grafička kartica. Uz to, postoji mogućnost ugradnje *PC Card* (PCMCIA) drajva u koji se mogu ubacivati sve standardne *Type II* kartice. Količina memorije koja se isporučuje u ovom kontroleru zavisi od zahteva kupca, a ograničena je isključivo brojem podnožja za SIMM module. Od kartica na raspolaganju su: ulazna sa 32 galvanski odvojena ulaza, izlazna sa 32 relejna izlaza - obe poseduju mogućnost rada na naponima do 220 V. Analogna ulazna kartica ima 16 ulaza sa 12-bitnom a/d konverzijom, koji rade na frekvencijama do 100 kHz. Analogna izlazna kartica ima osam 12-bitnih analognih izlaza maksimalne frekvencije 200 kHz. Osim ovih, postoje još i *counter timer* kartica sa četiri tajmera i PROMDISK kartica, koja služi kao zamena za hard disk u primenama gde vibracije ili druge osobine radne sredine mogu da spreče rad diska. Maksimalni kapacitet mu je 2MB i, osim memorije, ima ugrađen još i watchdog tajmer, jedan RS232 i jedan RS485 (ili RS422) serijski interfejs.



Drugi kontroler iz proizvodnog programa ove firme je *C3PO*. Razlika između njega i *R2D2* je u tome što se kod *R2D2* računar nalazi na zadnjoj ploči, dok je kod C3PO zadnja ploča pasivna, a računar se nalazi na kartici. Osim toga, *C3PO* ima mesta, pored procesorske i grafičke kartice, za još 4 kartica.

C3PO II je kontroler sastavljen od dva kontrolera tipa *C3PO* koji imaju mesta za po 7 kartica, ali mogu da rade u paraleli. Ovo je vrlo značajno u uslovima u kojima je potrebna maksimalna pouzdanost. U slučaju otkaza jednog od kontrolera, drugi preuzima sve funkcije i obaveštava nas o kvaru onog drugog.

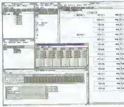
Još jedna firma kod nas se ozbiljno bavi industrijskim kontrolerima - u pitanju je *COMTEC* iz Novog Sada. Oni su se specijalizovali za proizvodnju PC kontrolera malih dimenzija, ali koji i dalje imaju kompletne funkcije velikih sistema. Što se velikih sistema i akvizicije podataka tiče, posvetili su se zastupanju firmi *Advantech* i *AEG Modicom*.

...e. Ali zbog specifičnosti industrijskih kontrolera (mala količina obrade podataka i neophodnost kontrole ulaza i izlaza) klasični programski jezici uglavnom nisu bili podesni. Zato su za programiranje industrijskih kontrolera razvijene razne tehnike, programski jezici i standardi.

Jedna od prvih i veoma široko rasprostranjenih metoda nastala je prilagođavanjem relejnih šema koje su se koristile i pre pojave industrijskih kontrolera. Ona je evoluirala u ono što danas zovemo *ladder* metodom programiranja ili *ladder* dijagramima. U ovom (uslovno rečeno) programskom jeziku programiranje se vrši crtanjem logičkih (relejnih) šema a zatim računar ovu grafičku reprezentaciju interpretira ili prevodi u mašinske instrukcije. Prednost ovog metoda je u njegovoj jednostavnosti tako da ga mogu koristiti i inženjeri bez ikakvog znanja programiranja. Ali da pri programiranju *ladder* dijagramima ne bi ograničili mogućnosti kontrolera moderni programi omogućavaju da se u *ladder* šemama dodaju i „uređaji" kakvi ne postoje u svetu digitalne logike. Tako možemo dodavati petlje, blokove sa malim programima, čak i aritmetiku pokretnog zareza.

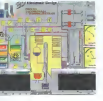
Za klasično programiranje industrijskih kontrolera razvijen je veliki broj specijalizovanih programskih jezika, od kojih je njih pet (SFC, FBD, LD, ST i IL) postalo industrijski standard. Ovi programski jezici najčešće su bazirani na *BASIC*-u (ali i drugim višim programskim jezicima) u koje su dodate naredbe za čitanje ulaza i manipulaciju izlaza i druge mogućnosti industrijskih kontrolera.

Osim ovih specijalizovanih jezika postoje i biblioteke funkcija za programske jezike C i C++ koje omogućavaju rad sa posebnostima" industrijskih kontrolera. I ovde se naravno postavlja pitanje standarda, jer nemaju svi industrijski kontroleri isti način adresiranja ulaza i izlaza. Taj se problem najčešće rešava tako što svaki proizvođač kontrolera uz hardver nudi i biblioteke funkcija. Međutim, adresni sistemi firmi *Burr-Brown* i *Intelligent Instrumentation* polako su se približili (prvenstveno zbog kvaliteta njihovih hardverskih komponenti) tome da postanu industrijski standard. Zbog toga danas većina proizvođača softvera za upravljanje industrijskim kontrolerima standardno podržava adresne sisteme ovih firmi.



Koncepcija po kojoj kontroler osim funkcije kontrole vrši još i vizualizaciju procesa promenila je pristup njihovom programiranju. S obzirom na to da su „novi" industrijski kontroleri kompletni PC kompatibilni računari na raspolaganju je velika količina softvera. Međutim, najbolje rezultate daju programski paketi koji objedinjuju funkcije programiranja i upravljanja.

Najbolji primer ovakvog softverskog paketa je *ISaGRAF*. Ovaj paket je obimno kompjuterski podržano softversko i inženjersko (CASE) okruženje. Dizajniran je tako da industrijske kontrolere pretvara u snažnu i potpuno otvorenu *PLC* arhitekturu. *ISaGRAF* koristi standardne industrijske *PLC* programske metodologije za kreiranje moćnih aplikacija bez potrebe za kompjuterskim jezicima. U sebi *ISaGRAF* ima ugrađenu podršku za rad sa grafičkim editorom za programiranje (sekvencijalni funkcijski tok, *ladder* dijagrami, funkcionalni blok dijagrami), tekst editorima (instrukcione liste, strukturni tekst, C kod i pet standardnih jezika) i dodatnim editorima (deklaracije promenljivih, definicije analogne konverzione tabele i I/O konfiguracije kao i debager). Pri tom se programiranje može obavljati na odvojenom PC računaru (sa *RS232*, *Ethernet* ili nekom dugom vezom između PC-ja i kontrolera) ili direktno na samom kontroleru (uz dodatak neophodnog monitora i tastature) ako on poseduje tu mogucnost. Da bi se povezala funkcija upravljanja i nadzora, *ISoGRAF* omogućava povezivanje sa najpoznatijim programima kao što su *INTouch*, *FactoryLink*, *Fix*, *Visual Designer* i *Genesis*. Kako *ISoGRAF* radi pod *Windows*-ima, ovo povezivanje je maksimalno jednostavno.



Od programa za nadzor procesa izdvojio bih *Visual Designer*. Ovaj paket koji radi pod *Windows*-ma, predviđen je za prikupljanje podataka, testiranje, merenje i kontrolu baziranu na PC-ju. Ono što predstavlja njegovu najveću snagu je potpuno intuitivni editor grafičkog prikaza sistema. On omogućava da se aplikacije prave bez i malo programiranja, jednostavnim crtanjem blok dijagrama ili *flowgrap*-a i određivanjem njegovih parametara. Time se dobija potpuno jasan grafički prikaz sistema sa svim potrebnim numeričkim podacima, kojima se sa jednog radnog mesta može upravljati i izuzetno složenim procesima.

Budućnost

S obzirom na sadašnje trendove razvoja celokupne kompjuterske industrije (jedan svet, jedna platforma, jedan operativni sistem) i sve prisutniju tendenciju da se kompjuter (makar mikrokontroler) stavi u svaki uređaj (od kućnih aparata do industrijskih mašina) možemo očekivati dva dominantna trenda u ovoj oblasti u sledećih nekoliko godina: maksimalnu standardizaciju i masovnu proizvodnju. Prvi će na udaru masovne primene kontrolera biti kućni aparati, automobili i takozvana zabavna elektronika. Uzevši u obzir masovno širenje PC računara u domovima širom sveta, za očekivati je da će svi kontrolisani uređaji u skoroj budućnosti biti povezani sa kućnim računarom ili će barem imati tu mogućnost. Za očekivati je i da će ti sistemi biti *Windows* kompatibilni. Ovo proizilazi iz činjenice da *Microsoft*, sem što proizvodi najbrojniji operativni sistem na svetu, jedini od proizvođača operativnih sistema ozbiljno planira svoj nastup na ovom tržištu. O mogućim posledicama ovakvog razvoja stvari i mogućim opasnostima koje on donosi u ovom trenutku ne vredi raspravljati. Ostaje nam jedino da čekamo i vidimo šta će se desiti. ■

Autor je diplomirani mašinski inženjer i urednik biltena Presek, glasila udruženja CADUA. Možete ga kontaktirati na e-mail adresu bobby.quyne@sezam.co.yu

NAŠIH 700 KORISNIKA

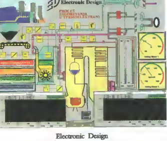


Elektrotehnički institut NIKOLA TESLA je sedamstoti korisnik naše opreme. PC kontroleri i sistemi za akviziciju podataka na PC platformama koju ED proizvodi više od 10 godina, koriste gotovo svi instituti, fakulteti i druge naučne ustanove kao i veliki broj preduzeća. Ostali proizvođači, trgovci i uvoznici zajedno, nemaju petinu ovog broja referenci. To je zbog toga što ED ima svoja optimalna i napredna tehnička rešenja realizovana na industrijskim standardima i modernom tehnologijom. Naši PC kontroleri i sistemi za akviziciju podataka imaju komparativne performanse sa najboljim proizvođačima ove opreme u svetu. Međutim, naši proizvodi imaju nižu cenu čak i od kineskih proizvođača.



Ovog meseca vam prestavljamo naš PC kontroler C-3PO II. To je dvo-procesorski redundantni industrijski PC sistem za akviziciju podataka, merenje i procesno upravljanje. Redundantna platforma je neophodna u primenama gde se od sistema zahteva velika pouzdanost. C-3PO II je istovremeno industrijski PC računar, veoma inteligentan PLC velikih mogućnosti, podstanica u mreži distribuiranog sistema (DCS), snažan RTU kao i centralizovan sistem sa jakim grafičkim interfejsom. Procesori među sobom rade preko komunikacionog modula ili preko dvostruke redundantne mreže (Eternet ili Arknet). Kontroler podržava rad sa standardnim diskovima, RAM/ROM disk emulatorima i sa PCMCIA. Podržan je rad sa saund blasterom za humane komunikacije i alarmna stanja. C-3PO II treba da zameni slične sisteme stranih proizvođača za više performanse, nižu cenu i PC kompatibilnost. C-3PO II ima garantni rok od 5 godina i servisni rok od 10 godina. Tehnička dokumentacija je na srpskom jeziku. Cena za početnu konfiguraciju (šasija sa dvostrukim napajanjem i bas modulima) sa softverskim drajverima i obukom je 1 995 bodova. Isporuka je odmah. ED na zahtev može isporuči kompletnu aplikaciju krajnjem kupcu.

Za C-3PO II postoji veliki broj softverskih alata i kompletnih paketa od domaćih i stranih kompanija.
Ako korisnik želi da napiše lično svoj softver na raspolaganju su mu softverski drajveri za DOS, WINDOWS i QNX operativne sisteme i to za C++, Pascal i Visual Basic. C-3PO II radi sa ISaGRAF paketom (IEC 1131-3 standard) koji se sastoji od programskih jezika kao što su leder logika, sekvencijalni funkcijski tok, strukturni tekst, funkcionalni blok dijagram i instrukciona lista.
Ako korisnik ne želi da uči programske jezike, može da projektuje svoj sitem sa generatorima aplikacija kao što su LABTECH CONTROL ili VISUAL DESIGNER.
Sa ovim, menijem upravljanjim softverskim paketima, moguće je napraviti aplikaciju, kao na slici, u roku od nekoliko sati. Cene su vrlo povoljne u odnosu na proizvode koje korisnik naručuje od neke specijalizovane softverske kuće i mogu se dati na zahtev.



Electronic Design

Electronic Design
Maršala Tolbuhina b.b. - Pejton
11000 Beograd Tel. 011/450-480 Tel. 011/444-74-95

Pogled u CAM

Nebojša Laz

Sistemi za automatizaciju proizvodnje polako, ali sigurno, nalaze svoje mesto u domaćim proizvodnim pogonima. Razmotrimo osnovne aspekte primene ovakvih CAD/CAM sistema, sa posebnim osvrtom na jedan domaći prozivod: NiceCAM firme CAD Professional. Možda još uvek imamo šanse da uhvatimo poslednji voz...

Razvoj savremene ekonomije i tržišta nameće određene principe i zakonitost kojima se svaki proizvođač mora prilagoditi. Efikasno prilagođavanje ovim zakonitostima je imperativ opstanka u uslovima rasta konkurencije koja se javlja kao posledica otvaranja sve većeg broja zemalja uticajima savremene tehnologije. Jedan od ovih principa, možda i najvažniji, je da cena proizvoda mora biti što niža. U današnjim uslovima konkurentna cena dobija se pre svega primenom visokoserijske proizvodnje. Međutim, sve veći rast konkurencije donosi i pojavu sve većeg broja različitih proizvoda, tako da proizvođači moraju biti u stanju da u što kraćem vremenskom periodu izbace na tržište noviji, savremeniji proizvod.

Najveću fleksibilnost u proizvodnji svakako poseduju ljudi, ali je njihova produktivnost mala. Osim toga, nove tehnologije zahtevaju i sve veći kvalitet proizvoda koji ljudi često nisu u stanju da postignu. Potrebno je, sa jedne strane, prilagoditi visokoserijsku proizvodnju stalnim izmenama proizvodnog programa, a sa druge, napraviti sistem za fleksibilan rad u malim ili srednjim serijama koje će biti isplative. Na taj način došlo se do tzv. fleksibilne automateacije, pojma koji predstavlja visoko automatizovani pogon, spreman da uz nikakve ili minimalne intervencije promeni kompletan proizvodni program.

Osnovna jedinica jedne ovakve linije je fleksibilna ćelija. Ona se sastoji iz nekoliko numeričkih alatnih mašina i jednog robota koji ih opslužuje. Ovakva ćelija sposobna je da radi u uslovima niskoserijske proizvodnje (sa stalnim izmenama asortimana proizvoda), a udruživanjem većeg broja njih u tzv. fleksibilne proizvodne sisteme dobijaju se i uslovi za rad u velikim serijama. O robotima i transportnim sistemima koji opslužuju fleksibilne ćelije može se pisati u nedogled, ali mi ćemo se zadržati na numerički upravljanim mašinama alatkama (NUMA) i razmotriti neke osnovne principe njihove primene.

Computer Numerically Controlled

Prvu ovakvu numerički upravljanu mašinu alatku predstavio je Institut za tehnologiju u Masačusetsu (MIT) 1952. godine, dakle u vreme kada su računari bili tek nešto više od dobre ideje. Prateći razvoj računara razvijale su se i mašine alatke, tako da se danas stiglo do savremenih CNC mašina (*Computer Numerically Controlled*) za obradu materijala, čije su upravljačke jedinice bazirane na sličnim (ili čak istim) mikroprocesorima kakvi se nalaze u kućnim računarima. Osnovni funkcionalni delovi ovakve mašine su: upravljačka jedinica (računar koji upravlja radom ostalih sistema), pogonski sistem (motori i reduktori) i merni sistem koji obezbeđuje povratnu spregu ka upravljačkoj jedinici. Na nekim mašinama postoje i mehanizmi za automatsku izmenu alata.

Ovakve CNC mašine imaju velike mogućnosti, ali je njihove računare skoro isto toliko nezgodno programirati na mašinskom jeziku kao i obične kompjutere. Da bi se pojednostavilo i pre svega ubrzalo programiranje CNC mašina, razvijen je veoma veliki broj viših programskih jezika (APT, EXAPT, NEL-NC, IFAPT itd.) koji su služili za automatsko generisanje putanja po kojima bi mašina trebalo da "prošeta" alat i na taj način obradi komad materijala. Projektant/programer više nije morao da raščlanjuje putanju alata na elementarne delove (prave i kružne segmente), već je jednostavno definisao površi kao takve (u parametarskom ili diskretnom obliku). Međutim, i dalje je morao da za dati predmet deli na površine, da izračunava njihove karakteristike, pravi transformacione matrice.

CAD/CAM

Sa druge strane, računari su poslužili kao sjajno pomagalo i pri procesu konstruisanja nekog predmeta. U tu svrhu razvijeni su mnogi CAD paketi, prvo za grafičke radne stanice, a potom i za kućne računare. Pod pojmom CAD-a podrazumeva se mnogo više od običnog prenošenja neke ideje na papir (računar). Savremeni CAD sistemi pored ovih poseduju i mogućnosti izvođenja različitih proračuna vezanih

Slika 1



Proces programiranja CNC mašine u klasičnom CAD/CAM okruženju

vodnje, predračun cene i sl.)

CAM sistema ima mnogo i uglavnom su rezervisani za brze radne stanice ali se u poslednje vreme mogu naći i moćni programi pisani za kućne računare. Situacija na ovom polju kod nas je u najmanju ruku žalosna. No, ako ste mislili da su CNC mašine neko čudo tehnike koje još nije stiglo u naše krajeve, varate se. Svaka veća fabrika ima priličan broj ovakvih mašina, a situacija u školskim ustanovama se iz dana u dan poboljšava.

Međutim, kada se radi o CAM sistemima, pitanje je da li je do skora postojao makar jedan ovakav paket koji je zaista bio upotrebljiv u proizvodnji. Srećom, situacija se i u ovoj oblasti menja na bolje, tako da se mogu naći primeri dobre integracije CAD/CAM sistema i u našoj privredi, ali je to i dalje na nivou pojedinačnih slučajeva.

a sam deo. AutoCAD je, na primer, krenuo ovim putem od kada je uveo solid modele. Nakon njihovog definisanja možete dobiti i podatke o masi dela, zapremini, položaju težišta, momentima inercije i sl. Neki CAD sistemi, pisani za projektovanje konkretnih delova i konstrukcija, poseduju i aparate za analizu naponskih stanja, proračun kinematskih i dinamičkih karakteristika i slično. Posledica automatizovanja ovakvih proračuna ogleda se pre svega u mogućnosti da projektant iterativnim postupkom, uz praćenje međurezultata mnogih pokušaja, dođe do finalnog rešenja koje će ispunjavati zahteve postavljene na početku procesa projektovanja.

Kao spona između projektovanja pomoću računara (CAD) i proizvodnje pomoću mašina kojima upravljaju računari, nastali su CAM sistemi. Pojam CAM (*Computer Aided Manufacturing*) označava skup programa koji služe, jednostavno rečeno, za automatsko programiranje CNC mašina alatki, robota, automatizovanih transportnih sistema i slično. Sada čovek koji želi da "oživi" CNC mašinu ne mora više da bude programer: ulaz u CAM sistem predstavlja crtež (i ostala tehnička dokumentacija) dela, a na izlazu se dobija program koji se unosi u memoriju upravljačke jedinice određene mašine na osnovu kojeg ona izrađuje odgovarajući predmet. Daljom integracijom CAD/CAM sistema sa raznim bazama znanja, ekspertnim i informacionim sistemima dolazi se do pojma računarski integrisane proizvodnje - CIM (*Computer Integrated Manufacturing*). Ovakvi sistemi automatizuju ne samo proces projektovanja jednog dela ili sklopa, već i celokupne njegove realizacije, uključujući tu i mnoge neproizvodne delatnosti (analize ekonomičnosti proiz-

Naše gore list

Firma *CAD Professional* iz Beograda (koja deluje u okviru firme *Digit*) našla je svoj interes popunjavajući ovu prazninu, i danas smo u mogućnosti da vam predstavimo već drugi njihov CAM paket. Prvi program, nazvan jednostavno *APT Generator*, bio je grafički interfejs za APT programski jezik, koji je radio u okviru *AutoCAD*-a. Glavni zadatak mu je bio da crtež koji je nacrtan u *AutoCAD*-u pretvori u APT program i prosledi ga APT procesoru iz kojeg se dobija standardan CL (*Cutter Location*) fajl sa opisom putanje alata. Ova CL datoteka se potom uvozila u postprocesor koji ju je prevodio u NC program prilagođen određenoj mašini.

Zadržimo se malo na ovom paketu i objasnimo neke osnovne pojmove u vezi CAM sistema. CAM sistem u svom okviru nema module za definisanje oblika radnog predmeta. To se radi u CAD programima, paralelno sa potrebnim proračunima, koji se zbog toga i povezuju sa CAM sistemima, tako da ćete mnogo češće naići na izraz CAD/CAM nego samo CAM. Neki CAM sistemi projektuju se kao nadogradnja na već postojeće CAD aplikacije, dok se drugi pišu ispočetka, tj. imaju sopstvenu CAD podlogu.

Pravi zadatak CAM aplikacija može se podeliti na dva dela: prvi je generisanje putanje alata na osnovu oblika radnog predmeta i zahtevane tačnosti izrade, a drugi je generisanje NC programa za konkretnu mašinu alatku na osnovu te putanje i zadatih tehnoloških parametara. U velikom broju slučajeva CAM aplikacije se i realizuju kao dva, uslovno rečeno, odvojena programa: procesora (za generisanje putanje alata) i postprocesora za "kompajliranje" te putanje u program razumljiv CNC mašini. Pomenuti *APT Generator* radi upravo na ovom principu: predmet se nacrta u *AutoCAD*-u, pozove se CAM modul koji na osnovu pomenutih podataka generiše običnu ASCII datoteku sa programom za izradu dotičnog predmeta pisanom u programskom jeziku APT (*Automatic Programmed Tool*), a ovaj se zatim prosleđuje APT procesoru - modulu koji na izlazu daje CL datoteku. U konkretnom slučaju iskorišćen je PERAPT procesor firme *Digital Research*. CL datoteka je ulazni podatak za postprocesor. Postprocesora ima onoliko koliko različitih mašina želite da programirate. Svaka mašina ima svoj jezik, i za svaku je potreban poseban postprocesor. Na kraju se dobija datoteka koja sadrži NC program, koji se putem RS232 protokola ili na neki drugi način može uneti u memoriju upravljačke jedinice mašine.

NiceCAM

Drugi CAM sistem firme *CAD Professional* kojem ćemo posvetiti naročitu pažnju jedan je od programa iz Nice paketa ove firme, u kojem se, između ostalih, nalaze i *NiceARC*, *NiceVIEW* i *NiceBASE*, moćni programi za arhiviranje tehnološke dokumentacije i najrazličitijih dokumenata u skoro svim grafičkim i DB formatima kojih možete da se setite. *NiceCAM* je realizovan kao *third party* aplikacija za *AutoCAD 13*. Izbor *AutoCAD*-a kao CAD komponente procesa proizvodnje ne treba mnogo obrazlagati: o njemu se mnogo zna i ubedljivo je najrasprostranjeniji CAD paket na PC računarima. Možda je verzija 13 malo neuobičajena za naše podneblje, ali to samo pokazuje ozbiljnost ove firme u pristupu problemu. Naime, i sam AutoDesk preporučuje da se sav dodatni softver za njihov program piše za verziju 13 (dakle i za *Windows*-e), da bi se na taj način postigla kompatibilnost sa narednim verzijama *ACAD*-a. Sa druge strane, *Windows* okruženje pruža udobnost o kojoj je mnogo toga već napisano. Ukoliko baš želite da radite pod nekom drugom verzijom, autori paketa obećavaju da će izbaciti odgovarajuće verzije svog programa.

NiceCAM je sistem namenjen programiranju CNC glodalica-bušilica sa 3 upravlji-

ve ose oakle kako za rešavanje konturnih tako za rešavanje površinskih problema. Ovim paketom možete raditi i tzv *point-to-point* programiranje (izvršavanje nekih operacija nad skupom tačaka pri čemu nije bitan način kretanja alata od jedne do druge tačke bušenje i njoj srodne operacije urezivanje navoja i sl.) ali za to postoje i jednostavni paketi manjih mogućnosti koji u potpunosti mogu da završe ovakve poslove. Postoji i mogućnost programiranja mašina sa čak 5 osa ali PC računari nisu najsrećniji izbor za takve probleme tako da je akcenat stavljen na 3D mašine što je po našem mišljenju više nego opravdano

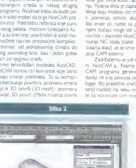
Najveća snaga ovog paketa skrivena je u svršeno iskorišćenim mogućnostima koje pruža *AutoCAD*. Proces projektovanja tehnologije za izradu nekog predmeta počinje naravno njegovim crtanjem u *AutoCAD*-u ili importovanjem crteža iz nekog drugog CAD programa. Predmet treba da bude površinski ili solid mobel da bi ga *NiceCAM* pravilno obradio. Prethodna rečenica krije punu snagu ovog paketa močnim funkcijama *AutoCAD* a za kreiranje površinskih ili solid modela možete nacrtati proizvoljno kompikovan predmet od jednostavnog cilindra do složenog avionskog krila kao i dobiti gotov program za njegovu izradu

Pored tehnoloških dodataka *AutoCAD* u *NiceCAM* donosi i tri komande koje bitno olakšavaju crtanje predmeta. To su komande za ofsetovanje površina, promenu smera polilinija ili 3D površi (*3D mesh*) i promena orijentacije 3D površi (MxN matrica površi tranformise se u ekvivalentu NxM). Koliko god da je složen oblik koji ste nacrtali *NiceCAM* će ga obraditi i dati vam korektan program za njegovu izradu. Filozofija kojom su se upravljali autori ovog paketa je da pri izradi programa za obradu nekog dela treba što više posla odraditi u samom *AutoCAD* u jer je on svakako moćniji od bilo koje *third party* aplikacije

Dakle nacrtali ste predmet koji želite da napravite i propustili ga kroz *NiceCAM*. Rezultat njegovog rada je putanja alata koja se vidi na ekranu i koja je potpuno ravnopravna sa svim ostalim entitetima kojima *AutoCAD* raspolaže. Putanju alata *NiceCAM* može da generiše na dva načina tako da se alat kreće u jednom smeru (*One direction*) ili cik-cak (*Zig-Zag*). Prvi način je pogodan ukoliko se obrađuje neki materijal poput drveta gde treba voditi računa i o smeru rezanja. Putanja alata je zapravo jedna velika polilinija koju možete do mile volje prepravljati pomerati rotirati skalirati ukratko sve što znate da radite sa polilinijama. U krajnjem slučaju mogli ste ovu putanju i ručno nacrtati i iskoristiti *NiceCAM* samo za generisanje NC koda (dakle ne i za generisanje putanja alata) ali se time negira razlog postojanja CAM sistema

Zadržaćemo se još malo na ovoj osobini *NiceCAM* a. Naime većina razvijenih CAM programa generisanje putanje alata stavlja na kraj procesa projektovanja tehnologije što praktično znači da ako u njoj (putanji) naiđete na neku neregularnost morate da ispravljate sam model a zatim da generišete novu putanju. U slučaju *NiceCAM* a to nije tako sve intervencije možete raditi direktno na putanji! Ova osobina paketa ima još jednu veoma korisnu primenu. Zamislite da želite da napravite program za finu obradu brodske elise (primer nije baš adekvatan za 3D glodalice ali poslužiće). Dovoljno je da nacrtate jedan krak elise, generišete putanju alata za njega i potom tu dobijenu putanju iskopirate u još dve zarotirane pod uglom od 120 stepeni. Naravno mogli ste i da onaj jedan krak iskopirate u još dva i da onda napravite jednu putanju alata ali to nepotrebno komplikuje, a i bitno usporava ceo proces

Slika 2



Nakon zadavanja osnovnih parametara putanje i izbora startne i krajnje (ulazne i izlazne) tačke alata NiceCAM generiše putanju glodala u obliku polilinije. Na slici je dat primer obrade jezgra za livenje kućišta ručnog fena od plastične mase. Može se videti način na koji se definiše konturna (2D) i površinska (3D) obrada

Kada ste završili sa eventualnim korekcijama putanje alata (koje su u normalnom radu sasvim izlišne) prelazi se na generisanje NC programa. Posle zadavanja nekih osnovnih tehnoloških parametara (brzine i smera obrtanja glodala veličine globala sigurnosnih rastojanja i sl.) i izbora tipa mašine (postprocesora) klikom na taster miša u trenutku dobijate gotov NC program! Taj program možete pregledati u posebnom prozoru (odličnom editoru koji nema veze sa *AutoCAD* om već je zasebna *Windows* aplikacija) pod nazivom *NicePAD*. *NicePAD* vam pored standardnih operacija koje ima svaki pristojan editor, pruža i mogućnost da putem RS232 interfejsa pošaljete program direktno u memoriju upravljačke jedinice određene CNC mašine i odmah je pustite u rad. Ne treba ni napominjati da možete podešavati sve bitne parametre komunikacije u zavisnosti od karakteristika RS porta ugrađenog u mašinu alatku

Možda će se nekom učiniti neobičnim što posebno naglašavamo ove mogućnosti ali treba imati u vidu da se kod nas i dan-danas veliki broj programa unosi u memoriju CNC mašina putem bušenih traka ili čak kucanjem. Koliko je to ekonomično i u skladu sa stavovima sa početka teksta - prosudite sami. Autori *NiceCAM*-a potrudili su se da i ovaj tehnički aspekt programiranja CNC mašina pojednostave što je samo još jedan plus celokupnom utisku o programu

U čemu je greška?

Svaki CAM sistem koji pretenduje na to da dobije atribut profesionalnog ne može se zamisliti bez mogućnosti naknadne provere ispravnosti generisanog NC programa. U skladu sa tim *NiceCAM* ima mogućnost da generiše putanju alata ne samo na osnovu predmeta koji je nacrtan u *AutoCAD* u nego i na osnovu već postojećeg gotovog NC programa. Sve što je potrebno je da se program koji se analizira učita u *NicePAD* izabere tip mašine za koju je pisan i - pritisne odgovarajuće dugme. *NicePAD* će koristeći mogućnosti koje pruža *Windows* okruženje za komunikaciju između programa proslediti *AutoCAD*-u putanju alata proisteklu iz analize datog NC programa i to opet u obliku polilinije koju možete proizvoljno prepravljati i eventualno generisati novi ispravan NC program

Simulacija kretanja alata i skidanja materijala nije ugrađena u ovaj paket ali osim lepšeg vizuelnog utiska takva simulacija ne bi donela ništa kvalitativno novo u odnosu na

izabrano rešenje. Ovim NiceCAM postaje zapravo dvosmeran sistem: možete na osnovu datog radnog predmeta izraditi putanju alata i NC program i obrnuto, na osnovu NC programa dobiti putanju alata. Koliko je ovo korisna osobina ne treba ni naglašavati: njome praktično dobijate mogućnost da analizirate i prepravljate veoma komplikovane programe koje je ko zna kad i ko zna ko napisao.

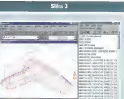
Pozovimo opet u pomoć onu brodsku elisu: zamislite da je pred vama program koji izrađuje takav jedan deo, ali vi hoćete da napravite istu elisu samo malo manju (ili veću). Ručno množenje svih koordinata određenim faktorom je blago rečeno smešno. I tu priskače u pomoć NiceCAM: iz postojećeg programa generišete putanju, skalirate je (u AutoCAD-u) i generišete novi NC program - sve ukupno posao od celih 5 minuta! Ovakva opcija ne krasi čak ni neke pakete koji su mnogo "profesionalniji" (i daleko skuplji) od NiceCAM-a.

Uporedimo sada osobine NiceCAM paketa sa nekim drugim CAM sistemima. Kao što se AutoCAD ne može porediti sa, recimo, IDEAS-om, tako se ne može ni NiceCAM porediti sa nekim visokoprofesionalnim sistemom poput CIMStation V. Uostalom, delovi koji se projektuju ovim veoma skupim programima daleko su složeniji od onih za koje je namenjen NiceCAM. Ako imate nameru da se bavite takvom vrstom posla, vrlo je verovatno da ni jedan program pisan za PC (pa i Macintosh) platformu neće moći da da zadovoljavajuće rezultate. NiceCAM je namenjen pre svega programiranju troosnih mašina i kao takvog ga treba i koristiti.

U poređenju sa drugim PC CAM sistemima NiceCAM ima dosta prednosti. Po našem mišljenju, najvažnije prednosti su generisanje putanje alata kao geometrijskog entiteta (polilinije) umesto u obliku CL datoteke, kao i minimalno vreme zadržavanja u CAM modulu. Kada kažemo da je putanja alata u obliku CL datoteke nezgodna (iako ovakav pristup problemu ima velika većina CAM sistema), to je pre svega zato što je ona sama po sebi nečitljiva, i da bi se prikazala na ekranu (radi verifikovanja) mora postojati zaseban program. Takođe, isključena je i mogućnost bilo kakve intervencije na njoj.

Svi CAD/CAM paketi trude se da unesu što više automatizacije i u sam proces crtanja radnog predmeta, pre svega pomoću biblioteka raznih standardnih elemenata. Tako smo imali prilike da vidimo programe koji se ponašaju "inteligentno", tj. imaju ugrađene algoritme za zaobljavanje oštrih ivica, grubu i finu obradu, ofsetovanje površina i kontura, popunjavanje "džepova" i ostrva i sl. Koliko god da je širok opseg "efekata" koji ovakvi programi pružaju, nikada se ne može napraviti sve za svakoga. U trenutku kada su se autori NiceCAM-a odlučili da za CAD podlogu uzmu AutoCAD umesto da razvijaju sopstven CAD podsistem, pravilno su zaključili da u tome treba ići do kraja. Na ovo smo mislili kada smo rekli da se projektant u CAM modulu zadržava što je moguće manje. AutoCAD je naime sam za sebe dovoljno moćan da mu nisu potrebna nekakva poboljšanja u stilu gorepomenutih; sve se to može uraditi u okviru AutoCAD-a na lakši ili teži način. Ukoliko ipak postoji potreba za nekom naročitom automatizacijom crtanja, tu je AutoLISP koji se savršeno uklapa u sve o čemu smo do sada govorili. Uostalom, da filozofija crtanja u AutoCAD-u nije dobra, on ne bi bio ono što danas jeste.

Slika 3



Na osnovu ranije ručno napisanog programa NiceCAM je u stanju da reprodukuje putanju alata i prosledi je AutoCAD-u. Nju je sada moguće prepravljati na razne načine i ponovo generisati NC program koji se pomoću NicePAD editora može poslati (preko RS232) u memoriju upravljačke jedinice mašine.

Pri analizi ovakvih CAM sistema naročito je interesantno obratiti pažnju na način na koji oni prevode krive površi u ravne segmente, i to pre svega iz razloga što se to direktno odražava na tačnost obrađenog dela. Poput CAD programa ni CNC mašine ne mogu baš "nacrtati" krivu liniju, bilo prostornu bilo ravansku, već je interpoliraju nizom pravolinijskih segmenata. Kada su u pitanju kružni lukovi u nekoj od osnovnih ravni (X-Y, Y-Z ili X-Z), ovu interpolaciju će izvesti operativni sistem upravljačke jedinice. Međutim, prostorne krive mašina jednostavno ne može da izvede i od programera očekuje da ih on ručno, ili češće pomoću nekog CAM sistema, razloži na niz duži.

Pristup CAM aplikacija ovom problemu je generalno gledano isti: krivu liniju (površ) CAD sistem razlaže na niz pravih segmenata i kao takvu je crta na ekranu. CAM sistem preuzima temena tih segmenata i kroz njih primenom nekog interpolacionog metoda provlači krivu, tačnije nalazi njen parametarski oblik. Zatim se ta kriva deli na niz tačaka i vrši obrnut proces: između svake dve susedne tačke provlači se duž, i takve duži se prosleđuju CNC mašini. Sva ova naizgled nepotrebna komplikacija izvodi se iz prostog razloga što je pri izradi krive konture (površi) u materijalu potrebno zadati daleko više ravnih segmenata nego što je to potrebno za pristojan izgled dela na ekranu.

Ova priča je naročito bitna kada se ima u vidu da se tačnost savremenih CNC mašina kreće oko jednog hiljaditog dela milimetra. U sistemima za programiranje CNC mašina nekim od viših programskih jezika (poput APT-a) tačnost obrade zadaje se kao i na tehničkom crtežu: procesoru (generatoru) se prosleđuju vrednosti tolerancijskih polja u okviru kojih treba da leži kontura, a on automatski izračunava na koliki je broj linijskih segmenata potrebno podeliti krivu da bi se dobila tražena tačnost. CAM sistemima koji poput NiceCAM-a rade u simbiozi sa nekim CAD sistemom direktno se zadaje broj ili dužina ovih linijskih segmenata. NiceCAM ne poseduje mogućnost ovakve interpolacije krivih linija. On će krive jednostavno podeliti na onoliki broj ravnih delova na koji je to učinio AutoCAD.

Ovo je na prvi pogled značajan nedostatak, jer se u samom AutoCAD-u svi elementi moraju deliti na daleko više ravnih segmenata nego što je potrebno za zadovoljavajući prikaz na ekranu, što dovodi do sporije regeneracije crteža. Međutim, treba imati u vidu da PC računari namenjeni ovakvim poslovima spadaju u kategoriju najjačih, tako da o sporosti prikazivanja crteža ne treba govoriti. Uostalom, jedan od osnovnih principa primene CAD-a kaže da je za kvalitet potrebna snaga.

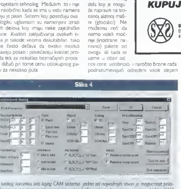
Već smo rekli da je jedna od velikih prednosti NiceCAM paketa nepostojanje CL datoteke već se na osnovu putanje alata direktno generiše NC kod za odgovarajuću mašinu Ovaj posao obavlja postprocesor koji je u ovom slučaju skriven od očiju korisnika. Međutim on mora postojati za svaki tip CNC mašine Stoga je u NiceCAM ugrađen jednostavan ali savršeno funkcionalan generator postprocesora U njemu se zadaju sve bitne osobine mašine i formata NC programa koji ona razume Ovaj postprocesor je dvosmerni što je još jedna od prednosti NiceCAM-a. Naime isti podaci o mašini koriste se i za generisanje NC programa iz putanje alata kao i za generisanje putanje iz NC programa o čemu je već bilo reči Pri instalaciji ovog softvera korisnik prvo definiše karakteristike svake od CNC mašina koje želi da programira i na taj način stvara malu biblioteku postprocesora koje poziva po potrebi

Pomenimo na kraju CAM sisteme koji pored navedenih osobina imaju ugrađenu logiku za podršku pri izboru tehnoloških parametara obrade nekog dela NiceCAM ovako nešto ne poseduje sve tehnološke parametre mora ručno da zada projektant-tehnolog Međutim to i nije ništa neobično kada se ima u vidu namena za koju je pisan Sistemi koji poseduju ovakvu logiku uglavnom su namenjeni izradi tipskih delova koji imaju neke zajedničke osobine Kvalitet zaključivanja ovakvih sistema je takođe veoma diskutabilan tako da se često dešava da ovakvi moduli olakšavaju posao i povećavaju kvalitet proizvoda tek za nekoliko beznačajnih procenata dižući po tome cenu celokupnog paketa i za nekoliko puta

NiceCAM je dobro zaokružen proizvod namenjen projektovanju tehnologije za izradu predmeta srednje složenosti najrazličitijih oblika i namene Poseban kvalitet daje mu jednostavno (skoro trivijalno) prilagođavanje konkretnom tipu upravljačke jedinice koju želite da programirate zahvaljujući ugrađenom generatoru postprocesora NiceCAM nije savršen ali se sa priličnom sigurnošću može reći da se uz njegovu pomoć na jednostavan način može projektovati proces obrade svakog dela koji je moguće napraviti na troosnoj alatnoj mašini (glodalici) Ne možemo reći da nismo videli moćnije (inostrane naravno) pakete od ovoga ali kada se uzme u obzir odnos cene udobnosti i naročito brzine rada podrazumevajući određeni visok stepen kvaliteta dobijenih proizoda NiceCAM ubedljivo tuče konkurenciju Ipak ono što je najlepše u celoj stvari je da je to domaći proizvod koji, samim tim ima obezbeđenu tehničku podršku ali i mogućnost prilagođavanja eventualnim specifičnim zahtevima korisnika Jednostavno rečeno CAD Professional se potrudio da napravi pre svega jedan upotrebljiv CAM sistem sa svim elementima koje takav jedan program treba da ima ■

Autora možete kontaktirati na e-mail adresu nlazic@sezam co yu

Slika 4



Za svakog korisnika bilo kojeg CAM sistema jedna od najvažnijih stvari je mogucnost prilagođavanja tog paketa konkretnom tipu upravljačke jedinice koja se programira U NiceCAM je ugrađen moćan generator dvosmernih postprocesora čime se omogućava programiranje praktično svih vrsta CNC glodalica

Korisna adresa

Digit & CAD Professional
Gradski park 2, Hala 'Pinki'
11080 Beograd-Zemun
Tel (011) 194-442 Fax (011) 612-784



BROJ 59 U ČITALAČKOM SERVISU



Predstavite se, molim

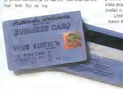
Potreba identifikacije ... kao i autorizacija pristupa određenim poslovnim sistemima prouzrokovala je ekspanziju uređaja sistema za identifikaciju (SZI). SZI predstavlja podsistem celokupnog sistema zaštite nekog poslovnog sistema. Osnovu svakog SZI čine identifikacioni dokument, uređaji SZI i institucionalna organizacija koja se brine o implementaciji ovog sistema u poslovanje korisnika.

ID kartica

Kao najčešći identifikacioni dokument najčešće se pojavljuje identifikaciona kartica (ID kartica). U svetu su najrasprostranjenije polivinilske ID kartice standardne veličine 54 x 85.6 mm. Zašto baš polivinil? Odgovor je jednostavan: taj materijal obezbeđuje trajnost kartice, otporan je na većinu fizičkih uticaja i oštećenja (grebanje, vlaga, sunce, savijanje itd.) a i njegova cena je niska.

Izdavalac kartice definiše se elementima na kartici koji su nepromenljivi i prepoznatljivi za sve kartice koje izdaje i deo su imidža izdavaoca, kao što su to zaštitni znak, boje, naziv, kodni sistem i dr. Shodno nameni i dizajnu, orijentacija kartice može biti horizontalna ili vertikalna. ID kartice mogu sadržati i dopunske elemente kao što su traka za potpis i embosirani tekst, a mogu biti i nestandardnog oblika. Korisnik kartice je osoba ili organizacija kojoj se daje pravo korišćenja ID kartice. Korisnik kartice nije, kao što se najčešće misli, istovremeno i njen vlasnik. Kartica je vlasništvo izdavaoca.

Kodni sistem ili šifarnik je sastavni i najvažniji deo SZI. On predstavlja skup određenih i unikatno pridruženih brojeva koji samo za izdavaoca kartice imaju funkciju, na primer redni broj, gradacija važnosti korisnika prema nameni, odobrenjima koje ima i sl. Kodni sistem najčešće je ispisan alfa numeričkom znacima, bar kodom, zapisom na magnetnoj traci i u novije vreme kompjuterskim zapisom u memorijskom čipu integrisanom na kartici, tzv. čip kartici.



Bar-kod sistem

Bar kod je sistem predstavljanja simbola putem razmaknutih linija različite debljine. Sistem izrade i čitanja bar koda je zbog svoje jednostavnosti veoma rasprostranjen. Čitač bar koda je u funkciji uređaja SZI. Postoje čitači sa vidljivim svetlom i infracrveni čitači. Postoji nekoliko osnovnih modela čitača.

- svetlosna olovka je najjednostavniji bar kod čitač, kvalitet se ogleda u uglu pod kojim olovka može da pročita kod
- stacionarni bar kod čitač na provlačenje kartice
- čitač bar koda sa udaljenosti može biti ručni (laserski) ili stacionarni (sreće se u prodavnicama), a udaljenost sa koje može da čita iznosi od 10 mm do 15 mm

Magnetni zapis

Magnetna traka je podeljena na tri dela koji se protežu celom dužinom. Zapis se vrši u svaku traku pojedinačno. U tabelama su dati osnovni podaci o trakama po standardu ISO 7811 2.

Uređaji za čitanje magnetnog zapisa su sastavni deo uređaja SZI. Oni su stacionarni, a mogu biti pisači, čitači ili kombinovano. Kodni sistem pripada specifičnim elementima sistema zaštite od zloupotrebe i falsifikata. Prilikom izrade kartice mora se obratiti posebna pažnja na zaštitu od falsifikata. Dizajn i tehnologija štampanja spre

TEMA BROJA: RAČUNARI U PROIZVODNJI

Sistemi za identifikaciju

	BPI	BPC	Br. karaktera	
T1	210	7	78+LRC	0-9,A-Z,. (ASCII 32-95)
T2	75	5	39+LRC	0-9,.,=,? (ASCII 48-63)
T3	210	5	106+LRC	0-9,.,=,? (ASCII 48-63)

BPI - bitova po inču, BPC - bitova po karakteru, LRC - karakter za kontrolu pri čitanju

čava da se standardnim postupcima uklone izmene ili dopune postojeći elementi ili unesu drugi koji ne pripadaju korisniku. Specijalni nanosi postavljaju se po potrebi preko bar koda radi vizuelne zaštite čime se onemogućava njegovo kopiranje i zloupotreba.

ID kartice čijom se zloupotrebom može prouzrokovati samo mala šteta dovoljno je zaštititi nekim jednostavnim i jeftinim postupkom. Međutim, kada se neautorizovanim korišćenjem ili falsifikovanjem ID kartice može naneti velika šteta izdavaocu ili korisniku, neophodno je obratiti više pažnje na zaštitu kako sistema tako i same kartice. U primeni je najrasprostranjenija hologramska zaštita - hologramski znak je specifična trodimenzionalna i višebojna slika koja omogućava vrhunsku zaštitu ID kartice od falsifikovanja. Dizajn holograma je izuzetno složen postupak, jer se pored boja određenih elemenata definiše i njihova dubina. Način na koji se dizajn kreira i realizuje u štampi je industrijska tajna poznata nekolicini proizvođača holograma u svetu. Kao takav on je izuzetno skup i isplativ je samo na velikom broju kartica kao što su to, na primer, bankarske kartice.

Ostali elementi i upotreba SZI

Pored ID kartice javljaju se i sledeći uređaji SZI:

- terminal za čitanje (registrator) može biti povezan sa centralnom jedinicom ili koncentratorom. Njegova funkcija je čitanje zapisa sa ID kartice, registrovanje i provera autorizacije korisnika i slanje informacija o samoj akciji (ulazak u preduzeće, dizanje novca u bankomatima i sl.) i vremenu njenog događanja.
- koncentrator prikuplja podatke od više registratora i prosleđuje ih na centralnu jedinicu. Koncentrator ima sopstvenu memoriju u koju može smestiti više hiljada podataka. On podatke linijski prima i prosleđuje, a memorisanje je u funkciji čuvanja podataka u slučaju nestanka struje ili neke havarije na SZI.
- podsistem za komunikaciju obezbeđuje vezu između uređaja SZI na različitim objektima poslovnog sistema i najčešće se naslanja na postojeći sistem PTT veza;
- centralni računar prikuplja i obrađuje podatke sa svih lokaliteta i uz pomoć aplikativnog softvera vrši njihovu obradu.

ID kartica odvojena od automatskog SZI označava pripadnost nekoj organizaciji čime je ujedno i reklamira u javnosti. U poslednje vreme uvode se i članske ID kartice za razne vrste klubova (restorane, video klubove, sportske klubove itd.), biblioteke, organizacije, udruženja i sl. Manifestacije kao što su npr. kongresi, sajmovi i seminari prosto žude za SZI. Njihovo uvođenje nije vezano samo za imidž organizatora, već postoji i njihova praktična upotreba (pogotovo na masovnim manifestacijama). Kartica standardno ima oznaku manifestacije, osnovne podatke o vlasniku i koristi neki od kodnih sistema. Najprikladniji je magnetni zapis zbog svoje mogućnosti da se smesti veliki broj informacija, ali i zbog niske cene celog sistema SZI. Posetilac se putem kartice identifikuje na određenim mestima u okviru manifestacije (u zavisnosti od šeme SZI).

Kod seminara i kongresa kartice imaju dopunsku ulogu da olakšaju komunikaciju između učesnika. Na taj način "Kako se zovete?", "Čime se bavite" i slična pitanja postaju suvišna. Velika preduzeća, bolnice i dr. pored obeležavanja svojih zaposlenih i posetilaca uvode i sisteme za registraciju pristupa i prolaza objektima i posebnim prostorijama povezanih sa praćenjem i kontrolom radnog vremena.

Osnovne karakteristike sistema za automatsku registraciju prolaza ili pristupa su brzina, fleksibilnost i ekonomičnost. Može se vršiti analiza i obrada radnog vremena ili nekih drugih parametara u zavisnosti od definisanih zahteva korisnika. Registracija prolaska se vrši provlačenjem ID kartice kroz čitač. Omogućena je registracija zaposlenih, privremeno zaposlenih radnika, sa radnika i posetilaca. Na osnovu plana rada preduzeća generišu se dnevni i zbirni izveštaji (ili neki drugi po potrebi) kao što su: ulaz, ulaz za prekovremeni rad, ulaz po nekom drugom osnovu, izlaz, službeni izlaz, privatni izlaz, izlaz na pauzu (i sl.), prisutnost po svim osnovama, zakašnjenja, odsutnost po svim osnovama (službeno, bolovanje, godišnji odmor, plaćeno odsustvo, porodiljsko odsustvo, vojna vežba i dr.), neregularni prolasci, svi prolasci, mesečni izveštaji o radu, ukupna prisutnost po svim osnovama, zakašnjenja, poimenična prisutnost, planirana odsutnost i dr.

SZI se prilagođava zahtevima korisnika i, s obzirom da radi pod personalnim računarom kao centralnom jedinicom, može da se integriše u jedinstveni informatički sistem preduzeća ili nekog drugog poslovnog sistema. U bankarskim sistemima kartica je nezamenljiva u negotovinskom plaćanju. Njene specifičnosti su u tome što se i izdavalac i korisnik kartice posebno štite od zloupotrebe sistema i falsifikata kartica. Zbog toga se njihova validnost verifikuje pre realizacije. To radi osoblje banke proverom ličnih dokumenata ili to automatski rade bankomati. Autorizacija svake izdate kartice kod bankarske kartice definiše se posebnim međunarodnim standardima za upis ličnog (PIN) broja na traci br. 1 magnetne trake i provere kontnog stanja korisnika. Kod čekovne kartice autorizacija se vrši uvidom u personalna dokumenta i neposrednim uvidom u tekući račun svakog korisnika.

Bez obzira na to kakav SZI bio i kakva mu je zaštita, da bi sve besprekorno funkcionisalo neophodna je korektnost njegovih korisnika. Tendencije razvoja SZI kreću se u nekoliko pravaca: Povećavanje autonomnosti sistema, integracija sa drugim sličnim sistemima, usavršavanje zaštite sistema od neautorizovane upotrebe i zloupotrebe, integracija sa video sistemima i usavršavanje ID dokumenata. ■

Autora možete kontaktirati na e-mail adresu: vlst@sezam.co.yu

	SS	ES	FS
T1	%	?	(
T2	[illegible]	?	=
T3	[illegible]	?	=

SS - startni karakter, ES - završni karakter, FS - separator polja unutar trake

Korisna adresa

BIROTEHNA A.D. Beograd
Bulevar Vojvode Mišića 37
tel. 648-699, 648-738

Softver je instrument!

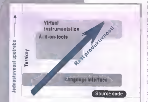
Branko Somardžić

Čini se da je, bar na ovim našim prostorima, upotreba PC-ja u svrhe akvizicije podataka još uvek u inicijalnoj fazi. Jedno je sigurno - bilo za testiranje, merenje, nadgledanje ili predikciju procesa, akvizicija podataka je nezaobilazni deo projektovanja sistema bilo kakve vrste!

Akvizicija (lat. *acquisitio*) označava pojam sticanja, prikupljanja ili akumuliranja nečega. Akvizicija podataka bi se mogla definisati kao aktivnost prikupljanja podataka o procesu koji analiziramo u cilju njegovog upoznavanja. Kao rezultat automatizacije akvizicije podataka izdvojio se zaseban segment računarske industrije koji se bavi proizvodnjom softvera i hardvera za automatizaciju prikupljanja podataka. Ovaj tekst ima nameru da ukaže na osnovne pravce razvoja softvera za akviziciju podataka (DAQ softvera - *Data Acquisition Software*), da pruži pregled osobina softverskih paketa namenjenih akviziciji podataka i da ukaže na neke zamke koje početnicima mogu doneti dosta glavobolja.

Kako je počelo?

Softver za akviziciju podataka evoluirao je na način sličan softveru opšte namene. U ne tako dalekoj prošlosti (pre petnaestak godina) softver za akviziciju podataka bio je strogo u domenu naučničke elite. Tada se pretežno radilo sa jezicima poput C-a, *Fortran*-a ili čak *assembler*-a. Napisane su linije i linije koda ekskluzivno za specifične aplikacije u kojima je najčešća hardverska platforma bio minikompjuter. Nešto kasnije pojavili su se specijalizovani računari namenjeni akviziciji (prikupljanju) podataka, ali je i dalje mnogo programerskog truda i vremena potrošeno za razvoj egzotičnog softvera za takve računare.



Danas je situacija bitno drugačija. Akvizicioni softver postao je pristupačan i veoma moćan alat koji bez problema mogu upotrebljavati i ljudi koji nisu eksperti u ovoj oblasti, a cilj svega toga je automatizacija prikupljanja podataka. Najnoviji paketi mogu simultano obavljati više akvizicionih funkcija i to sa veoma visokim performansama, čak i u okruženjima koja nisu naročito pogodna za insistiranje na brzini (npr. *Windows*).

Uprkos svim novotarijama i pogodnostima ugrađenim u akvizicioni softver, još uvek je velika umetnost odabrati pravu softversku podršku za sistem za prikupljanje podataka koji želimo da automatizujemo. Stoga svi koji žele akviziciju podataka kao sastavni deo svojih tehničkih aplikacija, moraju biti svesni mana i prednosti raznih softverskih paketa prisutnih na tržištu.

Potrebni uslovi...

Akvizicioni softver se koristi u aplikacijama poput automatizacije proizvodnih procesa, nadgledanja i kontrole procesa, u instrumentaciji, u testiranju elektronskih komponenti itd. Ipak, akvizicioni softver mora da zadovolji neke osnovne zahteve da bi uopšte mogao da se svrsta u ovu softversku kategoriju.

Pre svega, zahteva se **mogućnost kontrole hardvera za A/D i D/A konverziju** u punoj meri. To pretpostavlja visoki nivo fleksibilnosti kod procedura koje vrše identifikaciju i inicijalizaciju priključenog hardvera. Cilj je da korisnik ne mora da brine o hardveru za konverziju signala, nego da svoje napore usmeri na dobijanje željenih podataka i zaključaka koji iz njih slede.

Vizualizacija signala omogućuje prikaz signala na način blizak korisniku, veoma je pogodna za preliminarno procenjivanje kvaliteta prikupljenih podataka. Naravno, mogućnosti poput skaliranja dijagrama, zumiranja pojedinih njegovih delova ili prikaza više signala na jednom dijagramu su dobrodošle.

Srce svakog inteligentnog uređaja je algoritam koji dozvoljava izdvajanje željenih informacija iz podataka dobijenih akvizicijom. Stoga rutine koje daju **mogućnost procesiranja signala dobijenih akvizicijom** (spektralna analiza, digitalno filtriranje i sl.) predstavljaju srce softverskih paketa za akviziciju podataka.

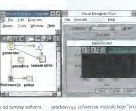
Prikaz prikupljenih i analiziranih podataka treba organizovati tako da se zahtevana informacija brzo i jasno stavi korisniku na raspolaganje. Tu se koriste razne tehnike kao što su alfanumerički prikaz, grafički prikazi i sl. Veoma važna osobina koju akvizicioni softver treba da poseduje je i **mogućnost memorisanja prikupljenih podataka** o procesu (u datoteku na magnetnom medijumu npr.). Ovakav pristup omogućava *off-line* procesiranje signala van realnog vremena, što je neophodno za obuhvatniju analizu prikupljenih podataka.

Na kraju treba pomenuti i **važnost postojanja interaktivnog i dobro opremljenog help sistema**. Naime, akvizicioni softver mora biti veoma jednostavan za upotrebu kako bi korisniku olakšao i ubrzao prikupljanje podataka. Stoga sve akcije koje korisnik treba da preduzme moraju biti logične, jasne i dobro dokumentovane u okviru help sistema.

Kategorije DAQ softverskih paketa

Do danas je akvizicioni softver evoluirao u pet potkategorija, od kojih svaka ima svoje specifične korisnike i aplikacije u kojima se primenjuje. Krenimo redom.

• *Turnkey* software je najjednostavniji za upotrebu i ne zahteva nikakvo programiranje. Sve što treba uraditi je kratko podešavanje (setup). Paketi ove vrste dizajnirani su da kontrolišu jednu fiksnu aplikaciju i to čine veoma uspešno. Problem je u tome što ne dozvoljavaju korisniku bilo kakve izmene (po pitanju hardvera i okruženja). Ova kategorija akvizicionog softvera po pravilu ima grafički interfejs uređen u vidu menija, putem kojih korisnik vrši potrebno podešavanje. Sama aplikacija je takva da najčešće na monitoru PC računara emulira neki realni merni instrument pomoću kojeg korisnik može obaviti željena merenja. Ovakav pristup ne zahteva nikakvo softversko razvojno okruženje (osim operativnog sistema) da bi se došlo do aplikacije. *DSPworks* firme *Momentum Data Systems* je primer turnkey softverskog paketa.



• Language interface software je potkategorija akvizicionog softvera organizovana u vidu biblioteka potprograma za popularne programske jezike opšte namene (*Pascal, Basic, C* i sl.). Za razliku od turnkey softvera orijentisanog ka korisnicima bez programerskih iskustava, ova kategorija softvera je namenjena korisnicima koji svoje aplikacije programiraju u konvencionalnim jezicima. Programeri u svojim programima pozivaju funkcije koje su sastavni deo jezičkog interfejsa, pa pre dobijanja izvršne verzije koda (sposobne za akviziciju podataka) treba izvršiti kompilaciju i linkovanje sa bibliotekama potprograma jezičkog interfejsa za akviziciju podataka. Olakšavajuća okolnost je što su rutine jezičkog interfejsa napravljene na dovoljno visokom nivou, tako da programer aplikacije ne mora da ulazi u detalje programiranja akvizicionog hardvera na najnižem nivou. Kada se podaci preuzmu iz posmatranog procesa i smeste u memoriju računara, nad njima se može primeniti uobičajeno programersko umeće da bi se oni analizirali, prikazali i/ili sačuvali za kasniju upotrebu. Jedan od često upotrebljavanih paketa tipa language interface je NI-DAQ paket firme *National Instruments*.

• Takozvani *add on tools* softverski paketi odlikuju se time što dopunjavaju poznati softver opšte namene (*Lotus 1-2-3, Excel*...) sposobnošću akvizicije podataka. Koristi su sasvim očigledne! Inženjeri i naučnici pošteđeni su programiranja u nekom programskom jeziku, rade u njima dobro poznatim programskim paketima opšte namene i pri tome imaju mogućnost obrade podataka dobijenih akvizicijom podataka! Primer softverskog paketa iz ove potkategorije je *Software Wedge* kompanije *T.A.L. Enterprises* iz Filadelfije.



• Popularni VBX-ovi (*Visual Basic custom controls*) predstavljaju softverske module koje "proizvode" pojedinci i firme u cilju dodavanja dodatnih funkcionalnosti *Visual Basic* razvojnom okruženju. U domenu akvizicije podataka treba ih svrstati u kategoriju add-on tools proizvoda. Razlikuju se od kategorije jezičkih interfejsa, jer ne predstavljaju biblioteke rutina za programske jezike opšte namene, nego su oni fajlovi koji unose mogućnost akvizicije podataka u *Visual Basic* razvojno okruženje.

• Kategorija source code softverskih paketa namenjenih akviziciji podataka odlikuje se time da su to uglavnom programi u vidu izvornog koda (namenjenog akviziciji podataka) koje je moguće uključiti u ciljnu aplikaciju i kompilirati zajedno sa ostatkom koda. Dobija se kompletna ciljna aplikacija koja ima sposobnost akvizicije podataka. *Source code* softverske pakete možemo tretirati i kao gotove primere koji pokazuju šta i kako treba isprogramirati na najnižem hardverskom nivou da bi odgovarajući hardver za akviziciju podataka "oživeo" u našoj aplikaciji. Korišćenje source-code paketa je verovatno do sada najkomplikovaniji način za programiranje sistema za akviziciju podataka, mada se njegove prednosti ogledaju pre svega u redukovanju ukupne veličine koda. Očigledno je da ovaj pristup zahteva poznavanje ne samo programskog jezika i akvizicionog hardvera, nego i intimnih detalja hardvera i sistemskog softvera PC računara! Ipak, u poslednje vreme ovaj pristup gubi oslonac u praksi, jer je novi hardver za akviziciju podataka veoma kompleksan i moćan u pogledu performansi i nameće niz veoma ozbiljnih pitanja koja se tiču interapata, DMA transfera podataka i sl. Tipičan predstavnik ove klase akvizicionog softvera je *DAQSourceCode* firme *National Instruments*.

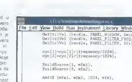
Virtualna instrumentacija

Virtual instrumentation software dozvoljava razvoj zamišljenih instrumenata koji se najbolje uklapaju u specifičnu aplikaciju za prikupljanje

pljanje podataka. Hardverska podrška ovoj veoma interesantnoj ideji su brojne plug-in akvizicione kartice koje se montiraju u slotove personalnog računara.

Tradicionalni instrumenti predstavljaju zasebne (*stand-alone*) jedinice koje se odlikuju mogućnošću priključivanja I/O signala, imaju fiksan "korisnički interfejs" a unutar uređaja specijalizovana analogna i digitalna elektronika prikuplja podatke iz procesa, analizira ih i na kraju prezentuje korisniku u pogodnoj formi. Problem je u tome što je uređaj, kada je jednom napravljen, neprilagodljiv raznim namenama, mada je to često potrebno. Nasuprot tradicionalnim, virtuelni (zamišljeni) uređaji koriste otvorenu arhitekturu PC računara, koja je postala industrijski standard, i u potpunosti zamenjuju svu komplikovanu elektroniku klasičnih instrumenata elektronikom akvizicionih kartica priključenih u slotove PC računara, dok procesorsku moć specijalizovanih mikrokontrolera preuzima CPU personalnog računara. Naravno, na raspolaganju su i memorijski resursi PC-ja, video sistem za prezentaciju korisničkog interfejsa, magnetni diskovi za trajno memorisanje prikupljenih podataka i, jednostavno rečeno, sve što današnji moćni PC računari mogu da pruže korisniku (multimedija, umrežavanje itd.). Na ovaj način postiže se cilj da funkcionalnost instrumenta i njegov korisnički interfejs budu potpuno podložni kontroli od strane projektanta aplikacije, a da je pri tome cena ovakvog rešenja mnogo prihvatljivija od cene čitavog skupa specijalizovanih *stand alone* uređaja koje bi bilo potrebno nabaviti za svaku od zahtevanih primena.

Danas dostupni softveri za akviziciju podataka, svrstani u pomenutih pet kategorija, nalaze se na raznim pozicijama u dijagramu jednostavnosti upotrebe u funkciji fleksibilnosti softverskog paketa. Međutim, današnji trend u razvoju softvera za akviziciju podataka usmeren je ka klasi softvera za virtuelnu instrumentaciju. Ova klasa odlikuje se najvećom produktivnošću, a da pri tome poseduje fleksibilnost (odlika source-code paketa) i lakoću upotrebe (svojstvenu *turn-key* paketima). Krajnji korisnik je taj koji definiše kompletnu funkcionalnost instrumenta, a pri tome cena koju plaća za komfor nije velika. Ne treba zaboraviti da je ovo dostignuće plod ogromnih investicija i tehnološkog napretka u okviru celokupne (kompjuterske) industrije.



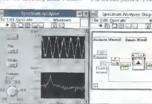
Šta još treba znati?

Raznovrsnost operativnih sistema u kojima akvizicioni softver radi predstavlja temu za dodatno diskutovanje. Čak iako je namera da originalna aplikacija bude napisana za jednu hardversku platformu (računar i prateći DAQ hardver), nije loše obezbediti da se aplikacija može izvršavati pod raznim operativnim sistemima prisutnim na tržištu, namenjenim datoj platformi. Na primer, ako je pre nekoliko godina aplikacija razvijana u *DOS* okruženju, danas će se sigurno pojaviti želja da se ona prenese i u *Windows 3.1* okruženje. Slično tome, ako se razvoj aplikacije radio u *Windows 3.1* okruženju, poželjno je da ona bude podržana i u *Windows-u 95*.

Potencijalni problemi mogu nastati i kod simultane upotrebe više različitih akvizicionih funkcija u jednoj aplikaciji. Činjenica da i hardver i softver imaju sposobnost analognog ulaza/izlaza nije garancija da će ove dve mogućnosti moći simultano da rade, a često i softver može biti ograničavajući faktor u maksimalnoj brzini akvizicije.

Takođe, tipična situacija u praksi je da se hardver za akviziciju podataka nabavlja od više proizvođača opreme. U takvom slučaju softver za akviziciju podataka koji podržava hardver raznih proizvođača se odlikuje ili otvorenom softverskom arhitekturom, ili ima apsolutnu prednost kod izbora! Naime, na ovaj način nismo ograničeni na ponudu jednog proizvođača opreme.

I na samom kraju jedan savet. Dobra je praksa da se pre dizajniranja aplikacije sagledaju i analiziraju razni mogući načini rešavanja problema (u domenu akvizicije podataka odgovaraju im razne "pozicije" na grafikonu jednostavnosti upotrebe u funkciji fleksibilnosti). Ovo podrazumeva analizu iskustava sa sličnim problemima, analizu (demo) varijanti DAQ softvera koji su potencijalni kandidati za implementaciju, konsultovanje kataloga više proizvođača DAQ hardvera i slične pripremne aktivnosti. Ovakvim pristupom mogu se uštedeti ogroman trud za razvoj aplikacije, značajna finansijska sredstva i osetno skratiti vreme potrebno za kompletiranje aplikacije. ■



Autor je saradnik grupe za instrumentaciju u okviru Laboratorije za elektroniku Instituta za nuklearne nauke "Vinča". Bavi se projektovanjem i izradom inteligentne merno-kontrolne instrumentacije, akvizicijom podataka i virtuelnom instrumentacijom. Možete ga kontaktirati na e-mail adresu brankos@rt270.vin.bg.ac.yu

Veliki plavi brat vas posmatra...

[illegible]šić

Članak koji je pred vama pokušaće da vam dočara kako vam jedan PC može pomoći da poboljšate i unapredite neki industrijski proces uz vrlo mala ulaganja. Zbog ograničenog prostora on vam, na žalost, ne može dati odgovore na sva pitanja, ali ako ste početnik u ovoj oblasti - a članak je upravo vama namenjen - mislim da vam može dati ideju i dovoljno smernica kako da potpuno samostalno izgradite jedan efikasan i moćan sistem za nadgledanje i upravljanje.

Vremena kada se PC koristio samo za obradu teksta ili poslovne grafike daleko su za nama. Otvorena koncepcija koju je PC ustoličio, iako manjkava, vremenom se pokazala kao jedna od retkih koje su preživele silne smene tehnoloških generacija. Iskreno rečeno, sumnjam da su inženjeri "velikog plavog" (IBM-a), kada su na samom početku ponudili na tržište svoj prvi PC, uopšte bili svesni svega ovoga, no u svakom slučaju dugujemo im zahvalnost.

PC je sigurnim koracima krenuo da krči put slave kao jedan od prvih računara koje je (tada svakako prilično bogati) pojedinac mogao da poseduje velikog spektra primene, a pritom je mogao i da stane na sto. IBM je počeo da ubire plodove svoga rada. Onda se desio Daleki istok sa magičnom reči "kompatibilno" i desetine hiljada jeftinih PC-ja sa nalepnicom "*Made in Taiwan*" preplavile su svet. Ostatak ove priče vam je sigurno poznat, dovoljno je samo da zavirite u kućište vašeg ličnog računara.

Isprva, PC je bio stvoren za obradu teksta, poslovne grafike, vođenje nekih malih baza podataka i sličnog, ali vremenom su se apetiti prosečnog korisnika povećali, a on je počeo da se interesuje za šta bi još mogao upotrebiti svoj lični računar.

Nije prošlo mnogo vremena dok su počeli da se pojavljuju i razni dodaci, bilo za kućnu bilo za (polu)profesionalnu upotrebu, posve različitih namena, ali pretežno u vidu kartica koje je sam korisnik mogao da ugradi u PC (setite se starih desktop kućišta kako su se jednostavno i lako otvarala!) i koje su se najčešće isporučivale sa prikladnim softverom i uputstvima. Tako je započeo razvoj različitih kartica za obradu zvuka, digitalizovanih slika, modemskih kartica, mrežnih i multiport adaptera i mnogih drugih.

Istorija PC-ja - istorija kompromisa

Korisnici su prihvatili PC koncepciju otvorene platforme: "Napravite od mene ono što vam je volja i prilagodite me vašim potrebama, ja neću biti tako efikasan kao drugi, za te poslove specijalizovani računari, ali ću vas koštati mnogo manje, a možete me i menjati kako se vaše potrebe menjaju." Nedugo zatim pojavili se i prvi primerci industrijskih PC-ja. Iako dobro maskirani u masivna industrijska kućišta ("rack mount") i opremljeni robusnim i najčešće neuglednim tastaturama i monitorima, bili su to oni isti PC-ji sa početka ove priče, ali "ojačani" izvesnim hardverskim dodacima i softverom.

Na industrijski način

Ispostavilo se da je PC, zbog svoje otvorene koncepcije, idealan za primenu u procesima kontrole/upravljanja, i da je efikasniji nego oni raniji usko specijalizovani računari. Za to je postojalo nekoliko dobrih razloga.

Prvo, konkurencija proizvođača PC opreme učinila je da cena proizvoda bude mnogo niža nego cena zatvorenih računarskih sistema. Drugo, time je i cena održavanja bitno smanjena, a održavanje pojednostavljeno tako da ga može obavljati i običan prosečno tehnički obrazovan korisnik PC-ja. I na kraju, svaki PC se može upotrebiti kao razvojna platforma za industrijski softver, što drastično smanjuje cenu razvoja i povećava efikasnost samog softvera. Nije potrebno ni pominjati kako PC, sam po sebi, daje daleko veće mogućnosti programiranja nego što je to slučaj kod starijih sistema. Ukratko, dobija se sistem koji je daleko bolji, fleksibilniji i lakši za održavanje, a istovremeno i mnogo jeftiniji nego što je to pre pojave PC-ja bilo moguće.

Kako dobiti industrijski PC?

Ovo je jednostavno pitanje, na koje nije tako lako dati prost odgovor: dosta zavisi od toga na kojem nivou hoćete da pristupite ovoj oblasti. Naravno, najjednostavnije je kupiti PC od neke za to specijalizovane firme, potpuno konfigurisan za predviđene poslove - dobićete ga u nekom prikladnom industrijskom kućištu ("rock mount" ili dr.) sa svim potrebnim ugrađenim i konfigurisanim hardverom, softverom prilagođenim vašim potrebama, tehničkom podrškom.

U ovoj varijanti ne morate da razmišljate o putu u inostranstvo - bićete prijatno iznenađeni ponudom i na domaćem tržištu. Ponuda je kvalitetna i na svetskom nivou, a što je najzanimljivije, po zaista pristojnim cenama.

U drugoj varijanti, dakle, ako vam nije previše bitan izgled dotičnog, već samo njegova funkcionalnost, a pre svega ako želite da u ovoj oblasti malo eksperimentišete, onda vam predstoji vrlo zanimljiv posao. Šta vam je činiti?

PLC? SCADA?

U drugoj varijanti korisnik mora prvo da definiše kakav je proces koji želi da kontroliše i/ili kojim želi da upravlja. No, pre nego što uopšte krenemo u razmatranje, moram vas upoznati sa dva termina

koji se pojavljuju gotovo na svakom koraku ka da se iole dublje zagazi u oblast zvanu industrijska primena računara.

• **PLC** je akronim od *Programmable Logic Controller* ili što bi se reklo po srpski - programabilni logički kontroler.

U suštini, PLC-om možemo nazvati sve - od jedne pločice koja ima EPROM i Spectrum-ov Z80 procesor a koja uz još par komponenti na sebi služi za kontrolu uključivanja/isključivanja neke mašine i sl. pa do npr. Pentiuma sa akvizicionom/upravljačkom karticom sa softverom u svojoj RAM-u i hard disku. Kako je ovo tekst o industrijskoj primeni PC računara, više pažnje u daljem tekstu posvetiću potonjim, iako i ovi jednostavni, od milošte prozvani "dummy" PLC, ovi imaju značajnu ulogu.

PLC je dakle "inteligentni" kontroler nekog procesa, tj. jedan računar sa svojim CPU-om, memorijom i programom koji taj procesor izvršava, i raznim kontrolnim ulazima i izlazima (analognim/digitalnim), već u zavisnosti od samog PLC-a i procesa. U svetu ih prave zaista mnoge firme, a ovom prilikom pomenuću *Allen Bradley-a, Omron, Electronic Design, Siemens* kao i *Westinghouse & Modicon*.

• **SCADA** je takođe akronim od *Supervisory Control And Data Acqusition*. Kada se pomene SCADA, misli se na ceo spektar raznog hardvera i softvera za prikupljanje informacija o nekom procesu, obrađivanja istih i u pojedinim slučajevima i reagovanja na adekvatan način.

Ako vam svi upotrebljeni termini nisu jasni - bez brige, biće o njima više reči kasnije. Sada ćemo da pokažemo kako sve ovo može da vama olakša život.

Kako?

Kao prvo, mora se odmah odrediti kakav je to proces kojim hoćemo da upravljamo ili da ga nadgledamo, koje su mu kritične, ključne i tačke na kojima se može vršiti povratni uticaj. Ovo možda zvuči kao da je istrgnuto iz udžbenika kibernetike, ali namera mi je da vas primoram da precizno definišete svoje potrebe. Kao drugo, morate znati sa koliko para i vremena raspolažete?

Što se prve stvari tiče, pogledajte da li je za vaš proces kritična temperatura, protok, pritisak, nivo tečnosti u nekom od sudova i sl. ili pak sve zajedno. Zaključujete da su vam potrebni senzori koji neku fizičku veličinu mogu da transformišu u odgovarajuću električnu. Uzećemo za primer hemijsko postrojenje u kojem je bitno da temperatura i pritisak neke hemikalije u jednom tanku budu konstantni. Kao otežavajuću (i uzetu samo primera radi) okolnost možemo pretpostaviti da tank ima neprekidan priliv a da odlivanje treba da se obavlja samo kada to dalji tehnološki proces zahteva. Kako temperatura i pritisak moraju da budu konstantni, u tanku mora postojati grejač koji će svojim uključivanjem/isključivanjem održavati temperaturu konstantnom, kao i dovod nekog gasa (vazduha npr.) iz kompresora, kako bi pritisak bio konstantan. Kao što se može videti, to je složen sistem, sa puno stvari o kojima se mora voditi računa, ali uz dovoljan broj pravih senzora i pomoć jednog prilično jednostavnog PLC-a njime se može vrlo lako vladati.

U ovom slučaju bili su neophodni senzori za očitavanje temperature i pritiska. Znamo da neki kristali reaguju na pritisak na taj način što im se na krajevima pojavi mala razlika potencijala, koja je proporcionalna pritisku (*piezo efekat*). Što se temperature tiče, postoje supstance koje menjaju svoju električnu otpornost u zavisnosti od temperature kojoj su izloženi (**NTC** i **PTC** otpornici), pa se uz njihovu pomoć vrši očitavanje temperature. Naravno, ovo je pojednostavljeno objašnjenje, ali korisniku je bitno da zna da postoje fabrički izrađeni senzori (radi ih i IHTM pri Prirodno-matematičkom fakultetu) raznih tipova i namene (svetlosni, temperaturni, za pritisak, za protok...) za koje je potrebno znati učvrstiti ih na mernu tačku i povezati.

Priključivanjem senzora uradili smo samo pola posla, jer ostaje i očitavanje. Logično, senzor ima i izlaze na koje "daje" određeni (najčešće) napon i time se njegova uloga u ovoj priči završava. Ali šta dalje sa naponom, jer je on, kao što znamo, jedna analogna fizička veličina, i kao takav nije prigodan za dalju upotrebu u svetu digitalne elektronike, kojem i pripada naš PC. Problem na pomolu! Ipak ne, jer ovde na scenu stupa priča o **A/D** (*Analogno/Digitalnoj*) konverziji. Time već polako ulazimo u svet PLC-ova i industrijskih PC-ja.

A/D?

Oni koji se danas bave, recimo, Hi Fi zvukom znaju koliko je Analogno/Digitalna konverzija (i Digitalno/Analogna, naravno) kompleksna oblast. Kada slušamo neki muzički CD ili muzičku potporu omiljene igrice preko *Sound-Blaster*-a najčešće i ne razmišljamo o tome kakav je put prošao zvuk od mikrofona do našeg zvučnika.

Zapravo, digitalni računari mogu da rade samo sa binarnim brojevima. Nužno je, dakle, neku analognu fizičku veličinu (u našem slučaju napon) na neki način pretvoriti u binarnu vrednost da bi se mogla dalje obrađivati.

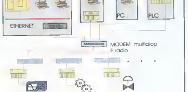
Kako? Potrebna nam je **A/D** (često se prave kao kombinacije, dakle A/D+D/A) **akviziciona** kartica ili **PLC** modul koji ima A/D konvertore. Kartica je najčešće veličine VLB grafičkih karti, sa specifičnim konektorom pozadi, i uz nju se najčešće isporučuje neki jednostavniji softver kao i uputstvo. Stavljanjem ove kartice u naš PC dobili smo već pola PLC-a. Ostaje nam samo da dodamo neku kontrolno-upravljačku karticu kojom ćemo moći da povratno reagujemo na proces, mada su integracije i upravljačkih funkcija na ove kartice vrlo česte.

Cene kartica su šarolike, od jedva stotinjak maraka za tajvanske, pa do nekoliko hiljada dolara za profesionalne kartice proizvođača kakvi su, recimo, *Burr Brown, Texas Instruments, National Instruments, Advantech* i drugi.

Naravno, **koliko** para - toliko i muzike, pa se uz ove tajvanske gotovo redovno isporučuju vrlo loša i siromašna uputstva na "Tarzan" engleskom, softver koji je najčešće prilično primitivan, a na eventualnu tehničku podršku jednostavno zaboravite, jer nigde nećete naći niti adresu niti faks proizvođača. Doduše, da ne "grešim dušu", ima primeraka koji su me svojim izgledom zadivili i koji stoje rame uz rame zapadnoj braći, ali po daleko razumnijim cenama. Međutim, ono što je najvažnije je da se sve ove kartice odlikuju određenim brojem analognih/digitalnih ulaza/izlaza, rezolucijom i brzinom uzorkovanja.

Broj ulaza/izlaza je maksimalni broj senzora i drugih srodnih uređaja koji se istovremeno mogu "nakačiti" na tu jednu karticu. On je najčešće zaista dovoljno veliki i kod onih najjeftinijih modela, osim ako nemate nameru da za predstojeće hladne zimske dane pravite svoju privatnu nuklearku i da je celu kontrolišete pomoću jedne jedine kartice.

Rezolucija se u video svetu upotrebljava kada treba opisati finoću tačke na ekranu, tj. posredno njenu veličinu. I ovde od rezolucije zavisi kolika će biti minimalna promena ulazne veličine koja će izazvati promenu izlazne u di-



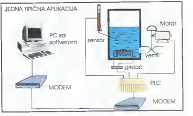
gitalnom obliku, recimo da li će 5,259 i 5,251 V nakon konverzije biti jedna ista ili dve različite digitalne vrednosti. Naravno, osim od rezolucije, ovo zavisi i od toga da li taj napon merimo u opsegu od 0-25 ili od 0-250 V i to je jedan od razloga zašto se rezolucija izražava u bitovima. A kad sam već pomenuo ove napone, da kažem i to kako je najbolje koristiti kartice čiji su ulazi/izlazi galvanski izolovani, i nije loše obratiti pažnju na ovo prilikom kupovine.

Ako je, na primer, rezolucija 8 bitova, znači da se neka analogna veličina na ulazu u datom trenutku može pretvoriti u jedan 8-bitni broj. A takvih brojeva može biti 2 na 8-mi, dakle tačno 256, odnosno u tom mernom opsegu se sa 8-bitnom rezolucijom može opisati tačno 256 različitih veličina. Da bi se izračunala najmanja merljiva promena dovoljno je podeliti raspon opsega sa 256.

Kolika vam je rezolucija potrebna zavisi samo od vašeg procesa, kao i samog senzora. Mada, pravilnim odabirom i mudrim raspoređivanjem može se i te kako uticati na to da vam i \"niskorezolucioni\" A/D konverteri \"odrade posao\", tako da ne morate da trošite pare na one skupe. Doduše, danas se sve do nekih 12 bita može dobiti po vrlo razumnim cenama.

I na kraju jedna napomena: u prospektima se retko nalaze podaci o tome koliko dotična kartica greši pri datoj konverziji, pa ukoliko vam je preciznost od velike važnosti, savetujem vam da se pre kupovine određenog modela ipak malo raspitate za iskustva (pre svega, negativna) kod nekog ko takav uređaj već koristi.

Učestalost uzorkovanja (sampling rate) je jedan od parametara akvizicione kartice, ali i softvera kojim se upravlja tom karticom. Izražava se u Hz (hercima) i označava koliko puta će u sekundi svaki od ulaza biti očitan, a njegova analogna vrednost konvertovana u digitalnu. Ako se bavite istraživanjima procesa u kojima je vremenski faktor vrlo bitan i kritičan (recimo, kod već pomenutih nuklearnih reaktora i sl.), tada već morate zahtevati veću brzinu uzorkovanja, mada ćete videti da i jeftiniji modeli pružaju sasvim pristojne i upotrebljive



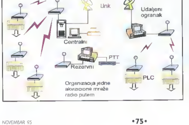
(bar za 95% primena) *sampling-rate*-ove.

Akviziciona kartica se najčešće isporučuje sa nekim jednostavnim softverom i naznakom da je kompatibilna sa većinom \"komercijalnog\" (hm, u našim uslovima vrlo rastegljiv pojam) industrijskog softvera na tržištu danas. Očigledno proizvođači se i ne trude previše oko razvijanja nekog pratećeg softvera, već upućuju na već postojeće softverske pakete, o kojima ćete nešto više pročitati na stranicama ovih \"Računara\".

Pomenuli smo da se A/D konverzija može odvijati i van industrijskog PC-ja, na nekom od za to podesnih PLC-ova. Rekli smo da i samih PLC-ova ima raznih, od samostalnih malih zatvorenih sistema koji služe za nadgledanje i upravljanje, do onih otvorenih, koji se preko (najčešće) RS232c serijske veze (ili preko bidirekcionog paralelnog porta, ili pak u PC svetu manje poznatih RS422 i RS485) priključuju na naš industrijski PC.

U ovom slučaju PC još uvek zasluženo nosi naziv industrijski, jer se na njemu vrti neki softver kojim se kontrolišu ti manje ili više nezavisni PLC-ovi (može ih biti i više), a tako posredno i sam tehnološki proces. On je naročito koristan u složenim tehnološkim procesima koji se odvijaju iz zasebnih tehnoloških celina. U tim slučajevima svaka celina ima \"lokalno glavni\" PLC na koji su dalje povezani senzori ili drugi PLC-ovi, i tako u dubinu sistemom podstanica. PC služi da se na jednom mestu mogu sagledati svi važni podaci vezani za proces, kao i da bi se sa jednog mesta moglo upravljati kompletnim procesom.

Ovi PLC-ovi mogu biti od jednostavnih od stotinjak (neretko i manje) dolara, pa sve do kompletnih PC (ili baziranih na recimo *Motorolinom procesoru*) računara od nekoliko hiljada dolara, sve u zavisnosti od potrebnih karakteristika. Prednost PLC-ova jeste u tome što svi gotovo po pravilu imaju relejne izlaze, direktno ili pak preko tzv. sklopki, vrše uključivanja/isključivanja velikih potrošača poput grejača, motora, pumpi i sl. Na taj način, u skladu sa primljenim informacijama, možemo povratno reagovati na tehnološki proces.



ATAME!

Postavlja se, međutim, pitanje kako sve to međusobno povezati? Sama veza između kontrolnih tačaka i centralnog PC računara je dosta važna. Mora biti dovoljno brza, a pre svega pouzdana i fleksibilna, tj. mora dozvoljavati da se kasnije lako može priključiti još neka merno/kontrolna tačka.

Uz senzore na dohvat ruke, nešto provodnika do akvizicione kartice odradiće posao na opšte zadovoljstvo. Problem je ako su senzori malo više razbacani: signal sa senzora je po svojoj prirodi analogan i osetljiv kada se \"transportuje\" na veće udaljenosti (max. od 2-3 do 10-ak metara). Međutim, vama treba *prava* vrednost i ne želite da gledate u pasulj koliko je signal na toj i toj dužini provodnika oslabio, zar ne? Jedno rešenje je da u blizini tih senzora postavite jedan jednostavni PLC sa A/D konverterom i RS232c na sebi, kojim ćete ga povezati na PC. Ovo je pogodno ako PLC ne mora da bude udaljen više od nekih 20-25 metara (zavisi od kabla, smetnji, kao i od kvaliteta samih RS232c PLC-a i PC-ja).

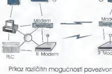
...ari su standardno opremljeni sa COM 1 i 2), mada se uz opasnost ...a oko deljenja IRQ a mogu doda ...dva (COM 3 i 4), što ipak ne pre... Pod UNIX-om ili nekim od njego... sa "multiport" adapterima koji su ...nski podržani ovakvih problema ...ma pa ako je nekom bitan broj a... možda je rešenje u UNIX u. Ali ...na vrlo bitan faktor?

...je situacija iz neke rafinerije gde se ...proces odvija na velikom prostoru i ...ohih međusobno zavisnih tehno... Nužno je imati jedan centralizo... za nadgledanje i upravljanje sa ko... svakom trenutku moglo očitati tre... bilo koje merne tačke (pritiska ...ne protoka...) kad i otvoriti ili za... voljni ventil ili uključiti pumpa i sl. ...lem su i daljina i broj.

...nekoliko protokola za serijsku ko... koji su popularni u industrijskoj pri... malo poznati prosečnim PC kon... su npr. RS422 i RS485 od kojih pr... va pouzdanu vezu do nekih 1200 ...brzini od oko 20 KB/s sa softver... ...lom veze (*handshaking*) a drugi br... 100 KB/s i još istovremeno veziva... ...ja na istu liniju (*point to multipoint*). ...zašto se RS485 često ugrađuje u sa... ...e tako da se nekad može iskoristiti ...ca na relaciji PC PLC pa onda dalje ...85 port samog PLC a.

...osim toga i takozvani *multidrop"* ...ne primene. No i sam PLC ovi ...rodni uređaji mogu biti dalje pove... ...pecijalizovanim protokolom koji je ...iran od strane samih proizvođača ...ovatno naleteti na povezivanja uz ...*bus-a*, *I(C bus-a*, **A-B DFI** i sl. Mo... povezivanja su kad što vidite ogranič... vašom maštom (budžetom).

...oslednje vreme pojavljuje se sve...vo i ...nje radio putem uz pomoć industrij... ...o-modema. Kao i većina stvari i ovo ...nje ima sijaset prednosti ali i jednu... Prednosti su očigledne: nema kablova ...a njihove instalacije i problema koji na... ...a se nečijom nepažnjom neki od njih



Prikaz različitih mogućnosti povezivanja udaljenih kontrolno / mernih jedinica

prekine ovako povezani uređaji su mnogo otporniji na atmosferska pražnjenja i sl. (iako to na prvi pogled ne izgleda tako). Osim toga ceo sistem se može lako proširivati i modifikovati jer nema neke određene fizičke hijerarhije podređenosti i nadređenosti već su sve tačke odmah i direktno dostupne. Uz pomoć jednog lap-top računara opremljenog radio modemom moguće je iz bilo koje tačke postrojenja pristupiti glavnom računaru i vršiti korekcije na njemu.

Takođe, lako se može postaviti rezervni računar koji prati razvoj situacije kao i onaj "glavni" i u slučaju kvara glavnog računara automatski preuzima njegovu ulogu i obaveštava (modemom ili zvučno) nadležne da je došlo do kvara. U slučajevima kada su tačke vrlo udaljene (i do nekoliko desetina kilometara, već u zavisnosti od konfiguracije terena) i kada se praktično ne isplati "vući" klasične vodove, ovi modemi su jedino rešenje jer se sa svojom cenom od 200 - 2500 USD (prosečno oko 600) i brzinama od 1200 do 9600 bps (komercijalni primerci) neuporedivo više isplate nego postavljanje bandera i kablova.

Naravno, imaju i "manu" a to je da je jedino za njihovo korišćenje potrebno odobrenje *Republičkog ministarstva za saobraćaj i veze*, odnosno potrebno je da imate dodeljenu frekvenciju za rad. U svetu je za industrijsku primenu dodeljen opseg frekvencija koji se može koristiti bez saglasnosti lokalnih organa, pod uslovom da je predajnik tačno određene snage i sa propisanim antenama. To je urađeno kako bi se omogućilo industriji da što pre primeni sva najnovija dostignuća iz ove oblasti a da je pri tome ne usporava administracija koja čak i na poslovično efikasnom Zapadu zna da bude spora. Ovakvo rešenje je vrlo pozitivno dokle god se obe strane drže dogovorenog, ali na žalost, ono za sada nije zaživelo i kod nas, mada postoje indicije da bi se nešto po ovom pitanju moglo promeniti pa treba sačekati.

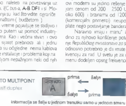
Sve u svemu...

Postavili smo na samom početku pitanje novca i vremena. Normalno uvođenje jednog ovakvog sistema košta i vremena i novca, ali ako ste ikada izračunali koliko košta vaš proces stoji dva sata ili čak čitav jedan dan, mislim da vam dalje ubeđivanje nije potrebno. Mislim da su vremena kada su se zastoj i veliki troškovi prihvatali kao nešto što je uobičajeno daleko iza nas i da se sada vodi računa o optimizaciji više nego ikada ranije. Automatizacija procesa nije trošak nego investicija i ako već raspolažete određenom količinom novca, u svakom slučaju ćete ih mudro "potrošiti" ako ih uložite u kupovinu ovakve opreme koja će sigurno sama sebe otplatiti.

Primeri sa Dalekog istoka najbolji su dokaz za gornju tvrdnju: tamo gotovo da nema procesa koji nije računarski kontrolisan i doveden do perfekcije. Kao rezultat dolazi nam proizvod koji po svojoj ceni može konkurisati bilo kojem sličnom u ostatku sveta. Naravno to ni tamo nije došlo preko noći, daleko od toga, trajalo je poprilično. Međutim, tamo je to shvaćeno krajnje ozbiljno i kao nešto u čemu leži spas i poboljšanje efikasnosti. Očigledno im je uspelo, pa zašto onda ne bi i nama?

A što se vremena tiče, da se opet poslužim jednom istočnjačkom poslovicom: "I put oko sveta počinje prvim korakom". Meni ostaje samo da vam poželim srećan taj prvi korak. ■

Autora možete kontaktirati na e-mail adresu sljubisic@sezam.co.yu



Pregled protokola za komunikaciju između udaljenih tačaka

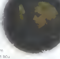

intel pentium
SUŠTINA JE NEGDE IZA ...
INFORMATION
SYSTEMS
tel. 011/650-506
tel./fax 011/681 949

A Brief History of Time



Ovo je elektronska varijanta istoimenog bestselera jednog od najvećih naučnih umova današnjice teorijskog fizičara Stivena Hokinga (Stephen Hawking). Mnogi ga porede sa samim Albertom Ajnštajnom zbog njegovog velikog dara za apstrakciju. Oblast kojom se bavi u ovoj knjizi je teorija relativiteta upotpunjena novim spekulacijama i saznanjima. Iako pogođen teškom i neizlečivom bolešću, Hoking poseduje sjajan spisateljski talenat, tako da su knjigu kupovali i (delimično) razumeli i laici.

CD izdanje koje predstavljamo ima znatno više informacija i crteža od papirne verzije, uključujući i brojne govorne zapise. Do svakog podatka se može doći na više načina, prateći logičan redosled izlaganja ili lutajući po želji kroz obilje materijala. Jednostavnost i intuitivnost je jedna od osnovnih odlika ovog izdanja.



Ako vas zanimaju najnoviji naučni pristupi večnim pitanjima koja se tiču suštine univerzuma, ovo je odlična prilika da zadovoljite svoju radoznalost. ■

Goran Alimpic



Dr. Ruth"s Encyclopedia of Sex



Sve što ste želeli da znate o seksu, a niste smeli da pitate' u računarski prilagođenoj formi mogao bi da bude podnaslov ovog CD izdanja. Doktorka Karola Ruth Westheimer je u Americi veoma popularna kao voditeljka kontakt programa i autor brojnih seksualnih priručnika. Na ovom CD-u nalazi se objedinjeno dugogodišnje iskustvo koje će mnogima biti od velike koristi. Radi se o veoma ozbiljnom projektu čija je glavna pažnja usmerena ka informativnosti sadržanoj u veoma detaljnoj i obimnoj enciklopediji i rečniku seksoloških pojmova.





Ilustrativni animirani prikazi seksualnih funkcija kod čoveka zajedno sa zvučnim i video odlomcima iz emisija dr Ruth čine glavni deo pomalo zapostavljene multimedijalne komponente ovog izdanja. Tu je i Seks kviz, igra koja je to samo na prvi pogled, pre bi se moglo reći da je u pitanju zbirka veoma korisnih vežbi za proveru znanja. Tu je i diploma u kojoj je opisano delo i životni put dr Ruth za one koji žele nešto više da saznaju o samoj autorki.

Glavna odlika ovog izdanja je informativnost – pregledne i dobro izložene činjenice imaju prednost nad samim načinom predstavljanja, tako da možete biti sigurni da ćete bez problema naći ono što tražite, i to najčešće u pisanoj formi. ■

Goran Alimpic

Objektno-orijentisano programiranje na jeziku C++

Dragan Milićev
Mikro knjiga Beograd 1995
470 strana



Vanredna knjiga ne samo za naše uslove. Pravo autorsko delo, a ne prevod druge (drugih) knjiga. Knjiga je pre svega svojevrsni referentni priručnik za jezik C++, jer najveći deo knjige objašnjava precizno i iscrpno elemente jezika C++. Praktično nikada nećete pronaći da u knjizi nema preciznog objašnjenja nekog od elemenata jezika, ma koliko složen za objašnjavanje on bio. Uvodni deo prikazuje osnovne principe objektno orijentisanog programiranja. Knjiga ne očekuje od čitaoca prethodno znanje jezika C, ali očekuje *kvalitetno* poznavanje bar jednog proceduralnog jezika (recimo, načina prenosa parametara, koncepta steka itd.)

Pisana za profesionalce, po svoj prilici preteška za čitanje potpunim početnicima, ova knjiga je ipak razumljivija od većine često otkranih knjiga o jeziku C++ koje su svesno "neodređenje" u opisima. Ne pretpostavlja da čitalac ima bilo koji određeni kompajler. U završnom poglavlju knjige predstavlja se opšta teorija objektno orijentisanog dizajna (nevezana za bilo koji određeni programski jezik). Neki detalji o jeziku se u ovoj knjizi pojavljuju prvi put kod nas. I sa tehničke strane knjiga je veoma kvalitetna, na šta nas Mikro Knjiga navikava svojim izdanjima. Grafička oprema knjige je izvrsna, možda najbolja za knjigu nekog našeg autora do sada.

Ako vas ozbiljno zanima C++, i sami ćete otkriti da je ovo korisna i kvalitetna knjiga.

Računari su o knjizi "pretpremijerno" pisali u jednom od prošlih brojeva u rubrici "Škola jezika C++".

Jonko Stamenović

OTVORENI SISTEMI - Koncepti, Komponente i Aplikacije za budućnost

mr Ejub Kajan
DIGP "Prosveta" Niš
358 strana



Knjiga je podeljena na tri dela: 1. Osnovni koncepti, 2. Komponente otvorenih sistema i 3. Aplikacije otvorenih sistema. Stil izlaganja je koncizan i jasan, razumljiv za svakoga ko raspolaže opštim znanjima iz informacionih i komunikacionih tehnologija. U prilogu osnovnom tekstu dat je spisak relevantnih standarda iz IT, rečnik skraćenica, otvoreni adresar, indeks autora i predmetni indeks. Knjiga je posebno značajna onima koji su odlučili da nastupajući milenijum dočekaju sa informacionim i komunikacionim tehnologijama, kako na radnom mestu tako i u privatnom životu.

U prvom delu autor jasno i precizno definiše koncept otvorenih sistema. Prema autoru, kada se karakteristike otvorenih sistema (portabilnost, skalabilnost (skalarnost) i interoperabilnost (interkomunikativnost)) nađu zajedno sa otvorenim međunarodnim standardima, sa mogućnošću da svako ko je zainteresovan može participirati i kada su rezultati jednako dostupni svima, onda govorimo o komponenti otvorenog sistema, a skup tih otvorenih komponenti čini otvoreni sistem.

U knjizi su obrađene sledeće komponente otvorenih sistema: hardverska platforma, operativni sistemi, otvorene komunikacije, sistemi baza podataka, grafički sistemi, korisnički interfejs i X Windows, programski jezici, administracija i zaštita.

U trećem delu obrađene su aplikacije otvorenih sistema, gde posebno treba istaći integrisani uredski sistem i CIM u otvorenom okruženju.

Knjiga se preporučuje svakome ko se bavi informacionim i komunikacionim tehnologijama.

mr Milan Milošević

PC95 - Windows 95, DOS 7.0, Hardver 95

Zoran Milosavljević, Dejan Ristanović, Zoran Kehler
PC PRESS
265 strana



Mlada izdavačka agencija PC PRESS munjevito je reagovala na pojavu *Windows*-a 95. Knjiga pod nazivom PC95 je prvo izdanje na našem jeziku koje se bavi novim operativnim sistemom. Sadržaj je podeljen u tri celine: u prvoj se opisuju instalacija, način rada u novom okruženju, elementi korisničkog interfejsa i standardne aplikacije koje se dobijaju uz *Windows* 95. U posebnim poglavljima posvećena je pažnja podešavanju sistema, radu sa štampačem i hard diskom, kao i razmeni podataka među aplikacijama.

U drugom delu knjige detaljno je opisan DOS u novoj verziji 7.0. Redom je navedena sintaksa i funkcija svake komande, uz dodatak koji se bavi *batch* programima. Treća i poslednja celina opisuje hardverske komponente računara sa posebnim osvrtom na njihovu efikasnost pri radu sa novim *Windows*-ima.

Knjiga je pretežno namenjena početnicima, tačnije onima kojima je *Windows* 95 prvi operativni sistem. Zbog toga su se autori trudili da koriste jednostavan i jasan rečnik, u čemu su uglavnom i uspeli. U toj funkciji je i dopadljiv i kvalitetan prelom sa mnoštvom ilustracija i primera.

Ovom izdanju bismo mogli zameriti nedovoljnu detaljnost u opisu iole komplikovanijih pitanja. Korisnik koji jednom savlada tehnike opisane u knjizi neće se često vraćati na deo koji opisuje grafički deo novog operativnog sistema, dok će mu neke napredne opcije ostati nejasne. Drugi i treći deo knjige su praktično modifikovane verzije umetaka objavljenih u starijim brojevima Računara. Dobro je, međutim, što se sve našlo na jednom mestu i što domaći korisnici najzad imaju priliku da se brzo obuče za rad na PC računaru pomoću jedne sveobuhvatne knjige.

Goran Alimpić ■

Objektno orijentisane knjige

Objektno orijentisano programiranje je poslednjih godina, zajedno sa C++-om, ušlo na velika vrata u svet softverskog inženjerstva i došlo već do takvog nivoa da je bez dobre literature sasvim nemoguće postići iole kvalitetniji rezultat. U ovom broju imate prilike da upoznate tri knjige koje spadaju u sam vrh svetske objektno orijentisane literature.

Žarko Berberski

Kad god neka tehnologija počinje da sazreva, postaje sve teže i teže u njoj uspeti bez mnogo čitanja i većeg broja knjiga. Kod objektno orijentisanog programiranja stvar je utoliko teža što ne postoji nikakav centralni autoritet koji bi rekao šta je glavni tok, a šta se može odbaciti.

Objektno orijentisani način rada i uzori za dizajn su čitav novi jezik komunikacije među programerima i oni koji to ne budu mogli da prate uskoro uopšte neće moći da komuniciraju i napreduju. Slična stvar se, zapravo, već desila prilikom prelaska na strukturirano programiranje - oni koji to nisu pratili vrlo brzo nisu mogli ništa više razumeti, kao što mi danas ne možemo da razumemo stare udžbenike za COBOL. Za ovih nekoliko godina koje smo proveli van sveta mnoge stvari su se promenile i sada možda prolazi poslednji voz nove softverske tehnologije.

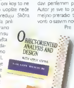
Jedna stvar je danas sigurna: C++ je dominantan jezik za objektno orijentisano programiranje (mada je C i dalje glavni svetski programerski jezik) i praktično je dobio svoj ISO standard (mada formalna procedura može potrajati i čitavu godinu). Sve četiri knjige koje ćete ovde upoznati koriste C++ kao osnovni jezik zapisivanja programskih primera. Druga stvar koja se može smatrati sigurnom je da sam programski jezik odlazi u drugi plan, a u žižu interesovanja dolazi metodologija stvaranja objektno orijentisanog softvera. Naime, glavni razlog uvođenja objektno orijentisanog programiranja je to da se utrošak vremena i energije predaci sa kodiranja na dizajn. Zato programer sve više postaje i dizajner i analitičar, a svaka greška u dizajnu se vrlo skupo plaća.

Postoji mnogo metodologija objektno orijentisanog dizajna i vrlo mnogo naslova raznih izdavača. Jedna je, međutim, priznata kao najcenjenija i najčešće pominjana, pa vam nju prvu i predstavljam.

Object-Oriented Analysis and Design

Grady Booch
The Benjamin/Cummings Publishing Company
1994, 608 str., 0 8053 5340 2, tvrd povez
cena: 34.95 USD

Ovo, međutim, nije ona stara knjiga od pre četiri godine koja je praktično započela objektno orijentisanu eksploziju. Ta knjiga je, i pored velikog broja citiranja, dobila zamerke da je malo suviše uopštena, da se mnogo bavi periferním programskim jezicima, itd. Autor je sve to primao i drugo izdanje temeljno preradio, tako da se skoro može govoriti o sasvim novoj knjizi.



Pre svega, sve u knjizi je u C++-u. Ako je u vreme pisanja prvog izdanja možda i moglo biti dileme ili barem priželjkivanja da neki drugi jezik bude glavni, u vreme pisanja drugog izdanja dileme više nije bilo: C++ je, zasnivajući se na popularnosti C-a, postao dominantan svetski jezik.

Autor je dopunio i pročistio svoju notaciju i učinio objašnjenja mnogo preciznijim i sistematičnijim. Zamerke na račun velike složenosti notacije rešio je tako što ju je razdvojio na osnovnu i naprednu. Osnovna će omogućiti čitaocima da brzo uhvate korak i počnu sami da je koriste, a onda će, kad im je i kako potrebno, usvajati napredne elemente. Moglo bi se ukratko reći da je autor prihvatio test realnosti i napravio dobar pomak od ponekad preterano akademskog izlaganja ka praktičnom, od kojeg će većina čitalaca imati neposredne koristi.

Verovatno bi autor i prvo izdanje drugačije radio, ali u to vreme je Booch-ova knjiga bila samo jedna od mnogih i niko nije očekivao da će se uopšte tako brzo izdvojiti kao knjiga koja će dobiti oreol glavne. To se možda u konačnoj liniji ne bi ni desilo da prošle godine nije izašlo ovo bitno poboljšano izdanje, sa notacijom koja je modifikovana u skladu sa zahtevima modernog objektno orijentisanog dizajna.

U te zahtave spada i to što ova knjiga sadrži čak pet kompletnih analiza slučaja - konkretnih projekata rešenih objektno orijentisanim metodom. Slučajevi obuhvataju veliki raspon od programa za automatsko prikupljanje podataka, preko biblioteke klasa, klijent/server arhitekture, pa sve do kontrole procesa i veštačke inteligencije primenjene ni manje ni više nego na kriptoanalizu.

Još jedan bitan pomak je od dizajna ka analizi. Kao i svaki programer, i autor ove knjige više voli da dizajnira nego da analizira (psiholozi kažu da zapravo generalno muškarci više vole da dizajniraju, a žene da analiziraju, no to je neka druga priča). Problem sa takvim stavom je to što se dizajniranje lako izvodi i sa nekoliko uopštenih pravila, ali njegov rezultat obično bude katastrofalan ako mu nije prethodila analiza.

Kad je objektno orijentisana tehnologija počela da se bori za primat sa relacionim bazama podataka, postalo je očigledno da joj nedostaju dobre analitičke metode. Te metode su još uvek daleko od celovitih, no ova knjiga je svojim pomeranjem ka analizi (koje u prvom izdanju praktično nije ni bilo) jasno pokazala da su i najpoznatiji autori shvatili njenu važnost i da čitava stvar ide napred. Već danas postavljeni metodi dovoljni su za sve probleme koji ne zahtevaju optimiziranje skladištenja podataka, a postoje novi pravci koji pokušavaju da glavne elemente relacione analize ugrade u objektno orijentisanu, no o tome drugi put.

Mada smo mi ovde još dosta daleko od velikih softverskih projekata, poglavlja posvećena procesu razvoja velikog softverskog projekta mnogo će pomoći programerima da dobiju kompletnu sliku, bolju perspektivu i veću efikasnost u toku rada na budućem projektu.

Za nestrpljive programere kojima je mnogo da čitaju celu knjigu o metodologiji postoji jedno prelazno rešenje koje se zove

Design Patterns

E. Gamma, R. Helm, R. Johnson, J. Vlissides
Addison Wesley
1994, 405 str., 0-201 63361-2, tvrd povez
cena: 36.95 USD

koje prava riznica isprobanih programerskih rešenja u okviru objektno orijentisanog di

zajna. Knjigu su potpisala čak četvorica autora, koje po mreži često u šali nazivaju "četvoročlana banda", mada se knjiga često citira samo po prvom autoru kao Gamma Et Al. Kako četvorica ljudi pišu knjigu? Kao što joj naslov kaže, ova knjiga je kolekcija uzoraka (autori je nazivaju katalogom) koji se direktno mogu koristiti u stvaranju C++ programa. Dakle, svaki autor je skupio i objasnio jedan broj uzoraka, a onda su to lepo sastavili u knjigu.



Knjiga je zapravo velikim svojim delom nastala na Internetu, gde su autori skupljali uzorke, diskutovali s ljudima, davali delove kataloga na čitanje i primali kritike i sugestije. Može se slobodno reći da bez Interneta ova knjiga ne bi nikada postojala, ako ni zbog čega drugog, a ono zato što je *Erich Gamma* u Kaliforniji, *Richard Helm* u Montrealu, *Ralph Johnson* u Ilinoisu, a *John Vlissides* u Njujorku. Zbog načina svog nastanka, ova knjiga je postala legenda još pre nego što se pojavila u knjižarama.

Svoju ogromnu popularnost *"Design Patterns"* duguje pre svega tome što daje kratka i efikasna rešenja za direktnu upotrebu u objektno orijentisanom dizajnu i to baziranom na C++ u. Kad knjiga sadrži 23 kompletno objašnjena uzorka sa C++ kodom, studiju slučaja i sažet pregled naprednih progamerskih tehnika, onda je to jednostavno knjiga koju programeri kupuju pre bilo čega drugog.

Sama tehnika korišćenja uzoraka (još nije teorijski zasnovana, a možda neće ni biti) preuzeta je od jednog arhitekte koji je daleke 1977, sa čak petoro kolega, napravio jezik uzoraka u arhitekturi i pokazao da postoji ogroman broj problema i rešenja koji se stalno ponavljaju i koji se mogu efikasno organizovati kao uzorci za kasniju višestruku primenu. Arhitekta *Christopher Alexander* je dokazivao da se čitave zgrade mogu konstruisati samo koristeći uzorke i nije mogao ni sanjati da će njegov rad poslužiti kao osnova za sasvim nov i vrlo efikasan pravac objektno orijentisanog dizajna.

Da ne bude zabune, uzorci nisu ni algoritmi ni tipovi podataka. Uzorci su modeli organizovanja i povezivanja klasa koje čine jednu operativnu celinu - baš kao što su fasada ili krov jedna građevinska celina, nekad veća, nekad manja, ali uvek konceptualna celina.

Autori su prikupljene uzorke podelili u tri grupe. Nazivi uzoraka će vam se ponekad učiniti čudnim i razuđenim, što je i za očekivanje, jer autori doslovno stvaraju novi jezik i uzimaju one pojmove iz raznih oblasti koji im se čine najplastičnijim za opis uzorka.

Uzorke kreiranja čine apstraktna fabrika, graditelj, metod fabrike, prototip i singlton (klasa koja može imati samo jedan primerak - objekt). Strukturalne uzorke čine adapter, most, kompozit, dekorator, fasada, laka kategorija i *proxy*. Uzorci ponašanja su najbrojniji i njih čine lanac odgovornosti, komanda, interpreter, iterator, medijator (posrednik), memento, posmatrač, stanje, strategija, metod šablona i posetilac. Budući da čak i najkraća objašnjenja svakog od ovih uzoraka zauzimaju dve strane teksta, o ovome možda u nekom od narednih brojeva.

Na kraju svoje knjige autori jasno kažu kako su svesni da je njihov katalog tek početak, a ne kraj priče o uzorima. Nastavak sledi godinu dana kasnije u knjizi

Designing Object-Oriented C++ Applications Using The Booch Method

Robert Martin
Prentice Hall
1995, 500 str, 0-13-203837-4, tvrd povez
cena 45.95 USD

i vodi C++ programera uporedo kroz C++ kod i dizajnerske dijagrame urađene *Booch*-ovom notacijom. Knjiga je zapravo sva sačinjena od studija slučajeva kroz koje se u hodu objašnjavaju i osnovni koncepti objektno orijentisanog dizajna. Međutim, težište je uvek na rešavanju konkretnog problema, često i do poslednje linije C++ koda.



Za programere koji smatraju da umeju lako i brzo da hvataju stvari u letu, knjiga sadrži kratak uvodni opis objektno orijentisane tehnologije, a zatim i objašnjenje *Booch*-ove notacije zajedno sa C++ kodom. U nekim slučajevima je zapravo prvo dat C++ kod, pa je zatim pokazano kako to izgleda na dijagramu.

Kroz sedam poglavlja autor polako pomera programera od kodiranja ka dizajnu. Tako se u prve dve trećine knjige nalazi mnogo izvornog C++ koda (preko sto stranica), dok ga u poslednjoj trećini praktično nema, jer se očekuje da je programer već dovoljno savladao notaciju i da može konkretna rešenja mnogo brže razumeti u dijagramima.

Studije slučajeva sa masivnim C++ kodom uključuju dosta poslovnih primena kao i pravljenje biblioteke klasa. Završna trećina, kao primer za analizu i uvođenje složenih rešenja (uključujući višeprocesni rad i kontrolu u realnom vremenu), koristi sistem za obezbeđenje upravljan računarom.

Ova knjiga svojom strukturom takođe sledi trend pomeranja težišta programiranja sa kodiranja na analizu i dizajn. Takav trend je već proizveo jednu novu, vrlo efikasnu tehniku rada - izdvajanje uzoraka koji se lako pamte i koriste, a u sebi sadrže ranije utvrđenu "mudrost", tj. proveren način rešavanja nekog tipiziranog problema. Time dolazim do treće knjige iza čijeg se skromnog naziva

Pattern Languages of Program Design

Uredili ***James Coplien i Douglas Schmidt***
Addison Wesley
1995, 574 str, 0-201-60734-4, meki povez
cena 38.95 USD

koju su kao urednici potpisali poznati *James Coplien* i manje poznati *Douglas Schmidt*. Knjiga je zapravo zbornik radova predstavljenih na prvoj godišnjoj konferenciji o jezicima uzoraka za programe (*Pattern Languages of Programs* - skraćeno *PLoP*).

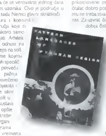
Dakle, za vrlo kratko vreme po izlasku prve knjige o uzorima, broj ljudi koji su time počeli da se bave toliko je narastao da je već održana i prva godišnja konferencija i objavljen zbornik radova s nje. Prva *PLoP* konferencija u Ilinoisu bila je u startu internacionalna sa priličnim brojem Nemaca i nekoliko Italijana.

Ukoliko ste navikli na to da su zbornici radova nabacana i nesređena gomila materijala koju je muka i pogledati - možete slobodno odahnuti, jer ovo je nešto sasvim drugačije. Pre svega, knjiga ima sređen grafički format i prostojno grafičko oblikovanje, tako da uopšte ne liči na zbor-

... već na koherentno delo. No to ... početak. Uzorci su naime takav inte... oblik koji od autora zahteva da uvek ... nešto konkretno, a opet dovoljno ... da bude široko primenljivo. Tako je i ova knjiga, mada formalno zbornik radova, zapravo još jedan bogat katalog uzoraka, ovog puta skupljenih direktno od autora iz celavog sveta. Donos osam oblasti ne samo primene, već i proučavanja i klasifikovanja uzoraka.

Radne okoline i komponente čine prvo područje, no ovde će verovatno sa razvojem biti još razdvajanja, jer su se sad u istoj grupi našli i interpretatori, i klijent/server okoline i uzorci za konstrukciju programerskih alata. Sistemi i distribuirana obrada su sledeće područje, koje uključuje protokole za međuobjektnu komunikaciju. Poslovne primene su se izdvojile u posebno područje i tu se pokazalo da se uzorci prilično lako hvataju u koštac sa bazama podataka i upitima. To je tek početak, ali je sasvim moguće da to bude lakši put za uključivanje dela relacione analize u objektno orijentisani repertoar.

Područje procesa i organizacije se, u stvari, malo naslanja na Booch ovo razmatranje procesa razvoja objektno orijentisanog softvera uz uvođenje analize. Ukratko, analiza počinje da ulazi i u područje uzoraka, ali ne kao obavezna faza, već kao jedan od uzoraka za rešavanje teških slučajeva. Zapravo, napravljeni su uzorci kojima se rezultati analize prevode u dizajn. Katalogiziranje uzoraka je oblast koja predstavlja prvi pokušaj da se izvrši sistematizacija i teorijsko zasnivanje čitave tehnike korišćenja uzoraka, koja će se verovatno jednog dana zvati teorijom uzoraka. Ovo je područje u kojem su za sada Nemci glavni istraživači.

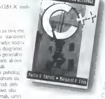


Arhitektura i komunik... su dva područja koja će verovatno uskoro samo stalno postojati. Arhitektura se ne odnosi na građevine, nego na softverske sisteme kojima se zbog raznih specifičnosti obično posvećuje posebna pažnja. Područje posvećeno korišćenju objekata i stilu ovde je okrenuto *Smalltalk* u korišćenju iskustava sa njime za opšte uzorke upotrebe objekata i organizovanja programa.

Još od vremena pojave *X Window*-a, pa preko *Macintosh*-a i *GEM*-a, programiranje zasnovano na obradi događaja pokazivalo je velike prednosti. *MS Windows* je obradu događaja po običaju "oteo" i učinio široko poznatom. Događajima i njihovoj obradi posvećeno je završno područje ove knjige.

Designing and Coding Reuseable C++

Martin Carroll, Margaret Ellis
Addison Wesley
1995, 350 str., 0 201 51284 X, meki povez
cena: 39,50 USD



Poslednja knjiga u izboru za ovaj mesec pristupa dizajnu sa stanovišta progamera koji mnogo radije kodira u C++-u nego da crta dijagrame. Ovakav pristup se danas generalno smatra zastarelim i opasnim, ali ova knjiga je briljantan izuzetak.

Poznati i rado čitani psiholog Erih From napisao je jednom da su knjige po svojoj prirodi dela koja ne mogu preneti živo iskustvo, već samo njegov mali, umrtvljeni trag. Povremeno se pojave knjige koje nekim čudom uspevaju da probiju pomenutu barijeru i ova knjiga je upravo od te retke vrste. Autori su jednostavno uspeli da slobodnim i živim jezikom, uz manje teksta i dosta primera, prenesu svoje živo iskustvo C++ programera, a da pri tome ne izgube na preciznosti, što se inače često dešava sa knjigama koje pokušavaju da budu žive.

Malo upozorenje nije na odmet: ova knjiga nije za početnike i one koji očekuju neprikosnovene definicije. Knjiga očekuje da čitalac dobro poznaje rad sa C++-om i da mu ne treba stvari crtati od praistorije do danas. Za uzvrat, čitalac otvorenog razmišljanja dobija čistu esenciju višegodišnjeg iskustva u formi koja se s lakoćom upija i čita u jednom dahu. Knjiga je prvenstveno posvećena stvaranju i korišćenju biblioteka klasa, jer su one glavni motor ponovne upotrebe koda. Brojne alternative razmatraju se do velike dubine i uvek sa otvorenim i razumnim argumentima za i protiv. Čitajući ovu knjigu, čovek jednostavno ima utisak da razgovara sa ljudima koji bez okolišanja prelaze na stvar i govore taman toliko koliko je potrebno da se jasno uhvati ideja i nt... rešenja ... ostalo su primeri i C++ kod.

Pokrivena su sva važna po... zajna klasa, proširivosti, efikasnosti ... ta, prenosivosti i kompatibilnosti, pa ... obrade grešaka, izgradnje hijerarhija ... mentovanja biblioteke.

Žene se retko pojavljuju kao autori računarskih knjiga i te knjige su često opterećene ne... ešto skr... enim na... ojem do... kazivanja ... da i ona to može, koji ... dobar ma... te al... teškim ... z... veštačenim... *Margaret Ellis* nema takvih problema. Postala je legenda kao koautor "*The Annotated C++ Reference Guide*-a", radila je u AT&T-ju na razvoju C++ prevodioca i to iskustvo i sigurnost se jasno osećaju u ovoj knjizi. Zanimljivo je i to da, izgleda, s lakoćom igra u autorskim tandemima, pa se mogu nadati da ovo nije poslednja knjiga iz njene tastature. Autori koji lako igraju u tandemima obično lakše i brže hvataju korak sa novim trendovima. ■

Zahvaljujemo se izdavačima što su nam izašli u susret poslavši nam knjige koje smo prikazali u članku.

Za više informacija

Addison-Wesley
Concertgebouwplein 25
1071 LM Amsterdam
The Netherlands

Napomena ...

Prentice Hall
Campus 400, Maylands Avenue
Hemel Hempstead
Herts HP2 7EZ
England
... 1442 881 9...

Alat za velemajstore

U poslednje vreme svedoci smo poplave integrisanih razvojnih okruženja. Za koje se odlučiti? Možda se u izboru isplati biti progresivan.

...a Kraljević

Sredinom 70-ih godina nekolicina stude...ata sa prestižnog MIT-a, nezadovoljna po...ećim alatima za razvoj aplikacija na relaci...m bazama podataka, odlučila je da napra... sopstveni alat kojim bi se olakšalo kreiranje ...državanje aplikacija i pritom obezbedila ...ksimalna portabilnost između različitih ...formi i operativnih sistema. Rezultat njiho... rada nije bio samo alat razvijen za jednu ...iku, već nešto što je moglo da posluži ...m krugu programera. Ubrzo su uvideli da ...rodaja alata profitabilnija od prodaje apli...ja. Tako je nastao *Progress Software Corpo...on* sa sedištem u Bedfordu, koji se nalazi u ...eričkoj saveznoj državi Masačusets.

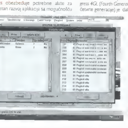
Meteorski uspon ove firme nije praćen ...esivnim marketinškim nastupom poput ...kurentskih firmi kao što su *Oracle*, *Ingres*, ...ose i drugi. To je ujedno i glavni razlog što ...zvodi *Progress*-a nisu toliko poznati na ...em tržištu. Međutim, o kvalitetu proizvo...ove firme svedoče kupci poput američkog ...istarstva odbrane, ministarstva poljopri...de, čuvene firme *Santa Cruz Operation* ...O), švedskog *Volvo*-a, nemačkog *Krupp*-a ...je ujedno i jedini proizvod oko čije super...e upotrebne vrednosti isto mišljenje ima...u konkurenti *Pepsi Cola* i *Coca Cola*.

...a je Progress

...gress obezbeđuje potrebne alate za ...pletan razvoj aplikacija sa mogućnošću portiranja na širokoj gami platformi i konfiguracija, host ili klijent/server okruženju, otvorenim (npr. Unix) ili zatvorenim sistemima (npr. VMS, BTOS/CTOS), sa karakter ili grafičkim korisničkim interfejsom i sve to bez promene izvornog koda. To omogućava programeru da se usredsredi na rešavanje konkretnog problema, a ne na rešavanje problema proisteklih iz promene konfiguracije računarskih resursa firme u kojoj ta aplikacija treba da radi. Svaka faza razvoja i korišćenja aplikacije zahteva određeno vreme i finansijska sredstva i teži se ka tome da se i jedno i drugo svede na što je moguće manju meru. Pritom takođe treba uzeti u obzir i postojeće resurse korisnika, tj. hardverski i kadrovski potencijal, da bi se minimizovala ulaganja potrebna za ostvarivanje projekta. Zahtevi koji se postavljaju pre izbora programskog alata su sledeći:

- Što kraći period razvoja aplikacije
- Što veća brzina i pouzdanost aplikacije
- Što lakše održavanje (faza održavanja je ujedno i najskuplja sa stanovišta programer-vremena)
- Što veća portabilnost (Iskorišćavanje postojećih resursa, povezivanje sa novouvedenom opremom)

Progress RDBMS (*Relational DataBase Management System* - Sistem za upravljanje relacionom bazom podataka) zajedno sa *Progress 4GL* (*Fourth Generation Language* - jezik četvrte generacije) je, dakle, sistem za razvoj aplikacija na relacionim bazama podataka. *Progress* je od početka osnivanja težio ka tome da izađe u susret ovim zahtevima, pa je shodno tome nastao programski alat koji ih u velikoj meri i ispunjava. Taj proizvod se vremenom unapređivao kako u interfejsu prema korisniku, tako i na polju bezbednosti i zaštite podataka i mrežnoj podršci.



Karakteristike

Da bi se stekao objektivni utisak o mogućnostima *Progress RDBMS*-a potrebno je navesti parametre limita koje taj sistem nameće. Prvi parametar odnosi se na veličinu baze podataka. Ona zavisi od veličine bloka u fajl sistemu na koji se *Progress* implementira. Veza između tih parametara data je u sledećoj tabeli:

Veličina bloka	Maksimalna veličina baze podataka
512 byte-a	*32 GB*
024 byte-a	*64 GB*
2048 byte-a	*128 GB*

Progress takođe ima limite za svaki od elemenata baze podataka, kao što su ukupna veličina fajla, broj i veličina tabela, broj i veličina sloga (*record*), broj i veličina polja u tabeli kao i indeksa, i svi ti limiti postavljeni su u zavisnosti od ograničenja operativnog sistema na koji se *Progress* instalira. U sledećoj tabeli nalaze se ograničenja za veličinu fajla nametnuta u zavisnosti od operativnog sistema.

Operativni sistem	Maksimalna veličina fajla
UNIX	*2GB ili manje u zavisnosti od sistema*
DOS	*8MB ili više uz opadanje performansi*
OS/2	*2GB*
VMS	*2GB*

Gore navedeni limiti prevazilaze se kreiranjem baze podataka u više volumena, tako da oni ne predstavljaju krajnje limite. Maksimalni broj volumena za jednu bazu podataka ide do broja 100, što znači da na većim sistemima možemo imati bazu podataka do 200GB. Integritet baze podataka garantuje se u potpunosti i to sa *roll forward*, *roll back* oporavkom od kraha sistema, sa dvofaznim izvršavanjem za distribuirane transakcije kao i sa *on line back up*-om za neprekidan rad.

U konkretnom primeru možemo posmatrati aplikaciju na DOS platformi rađi poređenja sa sličnim sistemima za rad u ma



njim i srednjim firmama. *Progress* sve podatke drži u jednom fajlu sa ekstenzijom .db (tabele, polja, indekse, validacione izraze itd.) i prilikom izvršavanja transakcija sve promene vodi u posebnom .bi (*before image*) fajlu. U slučaju kraha sistema nema potrebe za ažuriranjem indeksa, jer se u trenutku ponovnog startovanja aplikacije izbacuju sve transakcije koje su bile aktivne u trenutku kraha, te korisnik nastavlja sa normalnim radom bez nekih posebnih intervencija.

Takođe, programeru je data puna kontrola nad zaključavanjem zapisa u bazi za rad u mrežnom okruženju i to eksplicitno ili implicitno preko 4GL komandi za operacije nad slogovima u bazi. *Progress* ima striktno razdvojene operacije za rad na bazi podataka i mrežne protokole od aplikativnog interfejsa, tako da je omogućena potpuna prenosivost između različitih platformi i operativnih sistema. U praksi to znači da se baza podataka može nalaziti na *Unix* serveru, a da MS *Windows* klijentima pristupamo toj bazi preko, na primer, TCP/IP protokola, s tim da korisnik uživa u svim pogodnostima grafičkog okruženja, a da pritom zadržava prednost pouzdanosti i brzine *Unix* servera.

Progress podržava impozantan broj različitih platformi, a od operativnih sistema navodimo one koji se najčešće sreću: *Unix, VMS, BTOS/CTOS, DOS/Windows, Windows NT, System 7* itd. Od mrežnih protokola to su NetBIOS, TCP/IP, TLM, IPX/SPX itd. Trenutno je zaista teško naći konfiguraciju platforma/operativni sistem koju *Progress* ne podržava. Možemo reći da ono što *Oracle* ističe kao svoju prednost, *Progress* ima u potpunosti ostvareno.

Kupovinom razvojnog paketa kupac dobija mogućnost razvijanja i održavanja kompletnih aplikacija, a cena će zavisiti od toga na kojoj platformi je razvojna verzija predviđena da radi. Po završetku kreiranja i testiranja aplikacije, sve što treba videti je na kakvim hardverskim resursima i pod kojim operativnim sistemom ta aplikacija treba da radi, a onda uzeti *run-time* modul(e) (koje *Progress*, dakako, posebno isporučuje) za predviđenu konfiguraciju, te sve to fino upakovati i isporučiti. Dokupljivanje *run-time*-a se možda nekima neće svideti, pogotovo što se na našem tržištu to ne može nabaviti kod lokalnog pirata za budzašto, ali treba biti realan i shvatiti da je ovo alat za razvijanje kompleksnih aplikacija koje naručuju zahtevni kupci koji ne žele i ne smeju da prave kompromise. Kad se podvuče crta, nabavka *Progress* aplikacije po odnosu cena/perfomanse je mnogo povoljnija od konkurentskih proizvoda *Oracle* a, *Ingress*-a, *Sybase* i drugih. To je i razlog zašto po uglednom časopisu *"Var bussiness Report"* u kategoriji isporučioca alata za baze podataka nekoliko poslednjih godina *Progress* drži neprikosnoveno prvo mesto ispred mnogo zvučnijih i poznatijih imena.

Grafički korisnički interfejs

U poslednje vreme postoji tendencija ka korišćenju integrisanih grafičkih razvojnih okruženja. Svedoci smo prave poplave vizualnih alata poput *MS Access, MS Visual Basic* a, *MS Visual Fox Pro, MS Visual C, Borland Delphi, CA Visual Objects* itd. Svi ovi alati omogućavaju mnogo intuitivnije programiranje i većinom su namenjeni razvoju aplikacija koje rade pod grafičkim korisničkim interfejsom. Na polju obrade baza podataka najčešće pominjana imena na našem tržištu su *MS Access, Clipper, Fox-Pro, dBASE* (više verzija), koji su korišćeni za pisanje aplikacija namenjenih radu na PC-ju u manjim i srednjim firmama. U preduzećima sa većim zahtevima po pitanju količine podataka, brzine njihove obrade i broja korisnika koji imaju pristup, ta rešenja su uglavnom ostvarivana u *Oracle*-u, *Ingres* u, *Lynx* u, *Progress*-u i drugim alatima za razvoj aplikacija na relacionim bazama podataka.

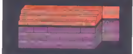
Progress je, poput ostalih značajnih proizvođača alatki na tom polju, takođe migrirao ka razvoju alata koji rade pod grafičkim korisničkim interfejsom. Počev od verzije 7, *Progress* je uveo značajne izmene u dotadašnji alat, kako na samoj bazi podataka, tako i na razvojnom alatu. Dodate su mogućnosti indeksiranja po rečima u karakter poljima, interne procedure koje se nalaze u samoj meta-šemi baze, a vezane su za transakcione glagole (*create, update, delete*), trigeri (okidači) koji startuju određenu proceduru u zavisnosti od toga da li su se neki uslovi ispunili prilikom transakcija na bazi i još puno čega.

Razvojni alat u grafičkom korisničkom interfejsu u verziji 7 naziva se *User Interface Builder* (UIB) i omogućava, poput svih ostalih vizualnih alata, kreiranje korisničkog interfejsa putem jednostavnih operacija dodavanja i premeštanja objekata po radnoj površini aplikacije. Dalja razrada je izlišna, jer bi nas odvela u detaljan prikaz razvijanja aplikacija pomoću *User Interface Builder*-a, što ostavljam za naredni broj našeg časopisa.

Treba napomenuti da se kao krajnji produkt kreiranja aplikacije dobija klasičan ASCII fajl (sa ekstenzijom ".w") koji možete ručno menjati po potrebi, za razliku od, recimo, *MS Visual Basic*-a, koji sve smešta u jedan .bin fajl nad kojim programer nema takvu kontrolu. Za rad na samoj bazi podataka koristi se *Data Dictionary* kojim se kreiraju ili modifikuju tabele, polja, indeksi, validacioni izrazi, trigeri itd. Osim UIB-a, u sklopu paketa alata dobija se i *Procedure Editor* koji predstavlja klasičan način za pisanje *Progress* programa, a pomoću njega mogu se editovati i fajlovi koje kreira UIB, kao i pisati programi koji su deo aplikacije, ali ne spadaju u korisnički interfejs, već u logički deo.

Kada se napravi aplikacija, programi se prekompajliraju i kao rezultat se dobijaju fajlovi sa ekstenzijom ".r" koji, dakako, nisu čitljivi za drugog programera, a mogu se i





dalje enkriptovat ove fajlove kor... ...d...ul. Kad je raz vije aplikacija potpuno je svejedno da li je razvijena u *Windows* ili *Motif* okruženju uz malo pažnje da se ne koriste specifični si stemski pozivi kao *Windows* DDE ili DLL. Sve grafičke objekte poput *combo box* ova selection lista *button* a slično *Progress* će sam prilagoditi okruženju u kojem se radi. Dakako ima tu još stvari kojima bi trebalo posvetiti pažnju kao što su *Report Builder* za brzo kreiranje formi i izveštaja i *RESULTS* za kreiranje korisničkih upita.

Sve ovo zvuči lepo samo da li radi? Radi. Od početka rada na razvijanju aplikacija u *ARES* u z Herceg Novog opredelili smo se za *Progress* iz svih onih gore navedenih razloga. Počeli smo na verziji 6.2 (razvojno okruženje sa karakter interfejsom) i dogurali do verzije 7.3B. Razvijen je veći broj aplikacija, al za primer bih uzeo paket *ARES R.A.M.* namenjen recepcijsko agencijskom poslovanju. Ovaj paket instaliran u Institutu Igalo uzet je za referencu jer su zahtevi za ostvarivanje aplikacije bili vrlo ngorozni. Aplikacija je instalirana u klijent/server okruženju gde server baze radi na *Windows NT* server mašini a radne stanice su sa WFW 3.11 kao i NT Client. Paket ARES R.A.M. razvijan je na *Windows* platform a sama baza i server su portirani na NT-u.

•••

Teško je u jednom članku sažeti ono što *Progress* nudi i dokumentuje u obimnoj literaturi ali se nadam da je to ipak dovoljno da se stekne prvi utisak o ovom alatu. Dakako najbolji način je sve to isprobati i potom doneti sopstveni zaključak. ■

Autor je konsultant za oblast računarskih mreža



Legenda živi

Jedan od najpopularnijih alata današnjice za rad sa bazama podataka pod DOS operativnim sistemom je svakako Clipper. Nova revizija (bolje reći verzija) ovog kompajlera donosi puno toga. Razvojno okruženje pod Windows-om, nove klase, alate i ideje ... Kakav će biti prelaz i da li se isplati?

Dejan Vesić

Prvi utisak koji je ovaj paket izazvao kod mene je "neobično i lepo". *Computer Associates* se ovoga puta stvarno potrudio. *Clipper 5.3* je izuzetno uspešan hibrid mnogih (proverenih) alata. No, krenimo redom. Sama

Instalacija

je laka i jednostavna. Ono što je neobično je da se odvija pod *Windows* okruženjem. Zbog veličine paketa (dolazi na 8 1.44 disketa) preporučujem da napravite direktorijume na hard disku DISK1, DISK2, ... i odatle uradite instalaciju. Zauzeće prostora na disku je pristojno za današnje aplikacije: oko 15 Mb. Ako paket instalirate na disku različitom od C: diska, proverite da li su SET naredbe u **AutoExec.Bat** korektne. *Install* program zna da se zbuni.

Odmah na startu valja razjasniti neke stvari: svi alati koji su postojali u prethodnim verzijama postoje i sada, ne morate menjati svoje navike i okruženje. No, ono čemu svakako treba dati šansu jeste novo razvojno okruženje, takozvani *Clipper* **WorkBench**.

Workbench

je ideja pozajmljena iz paketa **CA-VO** (*Visual Objects*, *Windows* alat za rad sa bazama podataka). Nije se puno promenio od beta verzije ovog paketa (detaljno predstavljene u "Računarima 105"). Prilikom startovanja paketa nalazite se u **Application Browser**-u: svaka aplikacija predstavljena je jednom ikonom. U jednom trenutku moguć je rad samo sa jednom aplikacijom. Rad je organizovan hijerarhijski (da ne kažem objektno...). Na vrhu je aplikacija, zatim moduli date aplikacije, i na kraju entiteti modula. Izbor između elemenata vršimo uz pomoć *Browser*-a. Prilikom otvaranja nove aplikacije, automatski se kreira modul **Binary Objects** - deo aplikacije koji će čuvati binarne objekte: menije, forme, opise polja i opise samih tabela. Svi ostali moduli imaju odgovarajuće .PRG ili .CH fajlove.

Entiteti su "najsitniji" elementi: procedure, funkcije i sl. Uz svaki entitet ide odgovarajući editor: za forme (odnosno izgled ekrana), za menije (izgled menija može se testirati iz samog *Windows* okruženja), *Field-Spec* editor (definicija polja) i *DB Server* (definicija tabela). Rad u nekom od editora obično za rezultat ima generisan kod u PRG/CH paru. Ozbiljna primedba *DBServer* editoru: ne može se raditi sa tabelama koje imaju memo polja u FPT formatu.

Tu je, naravno, i editor koda, koji ima standardne mogućnosti: CUA interfejs uz mogućnost kondenzovanog prikaza listinga (samo zaglavlja funkcija/procedura). Ostavljena je mogućnost i za eksterni editor (*CodeWright* je savršen izbor).

Prilikom definisanja nove aplikacije treba podesiti kompajler i linker setovanja. Uz linker opcije moguće je koristiti i tzv. *linker template*: svojevrstan skript jezik kojim je moguće u potpunosti upravljati procesom linkovanja, od izmene sitnih opcija do promene linkera.

BLOB polja

su nov mehanizam za rad sa velikim binarnim objektima (**Binary Large OBject**) kao alternativa standardnim DBT memo poljima. Uz *Clipper 5.3* dolazi **DBFCDX** drajver koji po *default*-u koristi *BLOB* mehanizam. Dobra vest je da se ovaj mehanizam može koristit uz bilo koji RDD drajver. **DBFMEMO** drajver sadrži samo delove za upravljanje memo poljima. Sve ostale delove za rad sa bazom (*Skip*, *Replace*, ...) treba naslediti iz nekog drugog drajvera. Primer za **DBFNTX**:

```
   REQUEST DBFMEMO
   RDDSETDEFAULT( "DBFMEMO" )
// Postavlja DBFMEMO kao default
   MEMOSETSUPER( "DBFNTX" )
// DBFMEMO "nasledjuje" iz DBFN-
TX RDD-a
```

I sam *DBFCDX* drajver realizovan je kroz nasleđivanje _**DBFCDX** drajvera od strane *DBFMEMO* RDD-a.

Ispod haube *DBFMEMO*-a nalazi se dobro poznati **FlexFile**. Ono što je u **SIX** drajveru bilo prisutno kao **VariField** tehnika, prisutno je i ovde: u memo polje moguće je smestiti promenljivu proizvoljnog tipa. Primeri kao što su:

```
   Replace MemoFld With 234
   Replace MemoFld With "Tekst"
   Replace MemoFld With Date()
   Replace MemoFld With .T.
   Replace MemoFld With
{1,2,3,{9,8,7}}
```

sasvim su regularni i otvaraju izuzetne mogućnosti (smeštanje kodnih blokova nije podržano). Tip promenljive u memo polju može se testirati sa klasičnim **ValType()** ali i sa novim **dbFieldInfo** (DBS_BLOB_TYPE, FieldPos(cFieldName)). Razlika je u brzini i načinu izvršavanja: *ValType* će učitati vrednost u memoriju i tek onda odrediti tip, dok *dbFieldInfo* radi to direktno, kroz ugrađene *FlexFile* mehanizme. BLOB polja su idealna za manipulaciju i prikaz slika ili bilo kojih "masivnih" podataka. No, ništa nije savršeno. Moguće je u memo polje učitati fajl proizvoljne veličine, dok će manipulacija sa njim biti u "komadima" ne dužim od 64Kb.

Tabele/indeksi

Sve BLOB funkcije se realizuju kroz niz novih funkcija koje daju/menjaju informacije o poljima, slogovima i samim tabelama. Tipičan predstavnik grupe je **DBINFO()**: možete saznati sve moguće: broj uspostavljenih relacija, niz zaključanih slogova, testirati integritet BLOB polja, pa čak pokušati i popravku oštećenog BLOB-a. Tu su još **DBRECORDINFO**, **DBFIELDINFO** i **DBORDERINFO**. U dokumentaciji se predviđa (buduća) mogućnost da se putem ovih funkcija promeni struktura objekta, a ne samo mogućnost da se informišete o njemu: za sada ni jedan RDD ne podržava tako nešto.

U *DBFCDX*-u je stari poznanik, **Comix** biblioteka, zadužena za rad sa indeksima. Ima sve što i konkurentni *SIX*: *Scope* (mogućnost da se slogovi filtriraju na osnovu vrednosti indeksa daleko brže i efikasnije od SET FILTER komande), izbor smera kretanja po indeksu (bez obzira na to da li je indeks kreiran sa *descend* flagom, moguć je rad u oba smera, funkcija *OrdDescend()*), zatim kretanje po indeksu kao da je kreiran sa *unique* klauzulom (uz otklanjanje svih problema koje je kreiranje takvih indeksa donosilo): kreirajte *tBrowse* objekat i u funkciji za skip funkciju *dbSkip()* zamenite sa *OrdSkipUnique()*. Za posebno specifične potrebe kreirajte "slobodne" indekse: dodavanje/uklanjanje ključeva je u potpunosti ostavljeno vama.

Klase

Dizajn *Clipper*-a je odavno objektni. Još od verzije 5.0 postoje ugrađene **tbrowse**, **tbcolumn**, **get** i **error** klase. U ovom izdanju je

novih klasa namenjenih izgradnji korisničkog interfejsa (**CheckBox**, **ListBox**, **TopBarMenu**, **PopUpMenu**...). Promenjene su postojeće klase, dodata je podrška za miša koja je (konačno!) integralni deo *Clipper*-a. Miš je transparentan kroz *Clipper*-ove sisteme (**READ/GET/TBROWSE**), naravno, imate i kompletnu kontrolu kroz za to predviđene funkcije. Možete ograničiti kretanje miša, paliti/gasiti/menjati mišji kurzor itd. No, najveće razočarenje je nemogućnost kreiranja sopstvenih klasa. Mada na tržištu odavno postoje biblioteke koje to rade (da pomenem samo **Class(y)**), u CA nisu smatrali za shodno da to omoguće kao standardnu opciju. Sa tekućom verzijom **Class(y)** biblioteke (2.0d) ima sitnih problema u radu, po *Clipper* kuloarima se šuška da će vrlo brzo biti na raspolaganju *patch* koji će probleme otkloniti.

Od "starih" klasa najviše izmena pretrpeo je *GET* sistem, sve u cilju što atraktivnijeg korisničkog interfejsa. Dodate su naredbe **@... GET CHECK/PUSH** (unos logičkih promenljivih), **@... GET LIST** (izbor iz predefinisanog skupa vrednosti), **@... GET RADIO** (izbor jedne od međusobno isključivih vrednosti). Pri tom je moguće zadati i neke grafičke elemente (izbor izgleda strelice i sl.), odnosno njihove tekstualne reprezentacije. Kao poslastica dodata je mogućnost da se preko **@... GET TBROWSE** omogući izbor iz tabelarnog prikaza (ovaj deo izgleda blago nezavršeno, potrebno je malo manipulacije sa metodama/varijablama da bi se dobio željeni efekat).

Light Lib

Jedno od vrlo prijatnih iznenađenja je i *Light Lib*, grafička biblioteka za *Clipper*. Da bi vaši programi dobili lep izgled, dovoljno je jedno *SET VIDEOMODE TO LLG_VIDEO_VGA_640_480_16*. Svi tekstualni elementi zatečeni na ekranu biće konvertovani u grafičke pandane. Sam *Clipper* 5.3 od video modova podržava još samo *LLG_VIDEO_TXT* (u dokumentaciji je ova konstanta pogrešno nazvana *LLG_VIDEO_TEXT*). U propratnom dokumentu LLIBG.TXT nalazi se narudžbenica za punu verziju *Light Lib* paketa koji podržava sve modove do 1280x1024 u 16 miliona boja. Podržani su prikaz slika u BMP i ICO formatu (**GBMPLOAD**, **GBMPDISP**), osnovne primitive za crtanje (**GELLIPSE**, **GFRAME**, **GLINE**, **GPOLYGON**, **GRECT**), a i rad sa fontovima (**GFNTLOAD/SET/ERASE**).

Fontovi su posebna poslastica. Moguć je rad sa dve vrste fontova: FND i FNT. FND fontovi su neproporcionalni video ROM fontovi fiksne veličine 16x8 tačaka. Koriste se kod svih standardnih *Clipper* funkcija za ispis. Sa druge strane, FNT su *Windows BitMap* proporcionalni fontovi u više veličina (4, 5, 6, 8, 10, 12, 14, 16, 20, 24, 30, 36, 40, 43). Naravno, omogućeno je i saznati širinu stringa bez samog ispisivanja na ekranu. Jedna seansa sa fontom bi izgledala ovako:

```
   Local aFnt, nColor  =3, nLen
Str
   Local cText   = "Hello World!"
   Set VideoMode TO LLG_VI
DEO_VGA_640_480_16
// Grafički mod
   aFnt =GFntLoad("ARIAL20.FNT")
        // Učitaj font sa diska
   nLen-
Str =gWriteAt(1,1,cText,LLG_MO-
DE_NIL) //Saznaj širinu u tačkama
   gWriteAt(1,1,cText,nCo-
lor,LLG_MODE_SET,aFnt)  // Ispis
teksta
   aFnt =gFntErase(aFnt)
   //Brisanje fonta i oslobadja-
nje
   // zauzete memorije
```

Uz sam *Clipper* stiže **CLIPPER.BML** (*BitMap Library*), biblioteka fontova i ostalih grafičkih resursa (npr. kurzori za miša). Tu su i alatke **BMLDIR** (pregled sadržaja BML fajla), **BML2RES** (vađenje fonta iz biblioteke) i **RES2BML** (dodavanje fonta/ikone u biblioteku). Spisak fontova je impozantan: 18 FND fontova (*Antique*, *BigSf*, *Block*, ...) i tri familije FNT fontova: *Arial*, *Times New Roman* i *Courier New* uz Small06.fnt. Da sve ne bude tako ružičasto, FND fontovi imaju samo velika slova USA abecede, dok FNT fontovi ne podržavaju YU slova (to niko nije ni očekivao, ali bilo bi lepo...).

Linkeri

Množina u podnaslovu je opravdana. CA nas je obradovao sa dva vrlo poznata linkera. Jedan je standardni **EXOSPACE** linker za programe koji će raditi u zaštićenom modu procesora. Razlike u odnosu na verziju 1.0g (koja je išla uz *Clipper 5.2e*) su minorne. Iz nekog razloga isključen je paket za podršku miša (*EXOSPACE PACKAGE MOUSE*). Kratko objašnjenje: za programe pisane u čistom *Clipper* kodu to nije ni potrebno, dobro dođe kod nekih dodatnih biblioteka koje rade na niskom nivou, tj. sa hardverom računara. Na BBS-ovima je odnedavno prisutan i prvi *update* za *Clipper 5.3*. Kao dodatak uz *update* dolazi i **OPTEDIT**, program kojim možete do vrlo finih detalja podesiti ponašanje vaše zaštićene aplikacije. Primenjuje se na generisani .EXE, možete podesiti način utroška memorije, svapovanje i korišćenje UMB-a.

Pravo (prijatno) iznenađenje je verzija **BLINKER**-a. Kažem "verzija" jer se ne radi o "pravom" *Blinker*-u, već o varijanti koja je uskraćena za bitne stvari: u dokumentaciji se na *Blinker* referiše kao *"Real mode linker"*. Uklonuta je podrška za zaštićeni mod, inkrementalno linkovanje i podrška ostalim jezicima/kompajlerima. Kao i uz LLIBG, postoji *Blinker.Txt* (odnosno WRI verzija) sa ponudom za upgrade na punu varijantu. Ono što je bitno: i bez inkrementalnog linkovanja *Blinker* je daleko brži od *ExoSpace*-a, tako da je savršen za razvoj programa. *ExoSpace* možete koristiti kao finalni linker. Naravno, nema nikakvih problema da umesto ovog *Blinker*-a koristite puno izdanje (tekuća verzija 3.20) ako ga imate. Time koristite ono najbolje od oba sveta: zaštićeni mod, inkrementalno i izuzetno brzo linkovanje. Uz pomenute *template* ove za *Workbench*, možete *Blinker* proglasiti jedinim linkerom za vaše okruženje, što vam toplo preporučujem.

Ostalo

Extended *sistem je proširen sa nekoliko novih grupa API funkcija, da bi podržao LLIBG i* ExoSpace *podsistem. RDD podsistem je drastično unapređen, tako da iz C-a možete napisati aplikaciju koja će se koristiti samo slojem za rad sa tabelama. Dodate su neke davno potrebne funkcije: manipulacija direktorijumima* (**DirChange**, **DirRemove**, **DiskChange** i **MakeDir** - u dokumentaciji se na nju pogrešno referiše sa **DirMake**). Često se pozivam na dokumentaciju, a ista dolazi u dva formata: uobičajene *.NG datoteke i *Windows* HLP fajl. Ona je najslabija tačka paketa. Stiče se utisak da je sastavljena na brzinu i da mnogih bitnih stvari nema. Sve funkcije i naredbe su tu, ali nedostaje operativnost, razlike u odnosu na prethodnu verziju i opis bitnih novih stvari koje 5.3 donosi. U sledećoj iteraciji biće bolje.

Zaključak

Od kada je CA kupila *Clipper*, on grabi divovskim koracima napred, svaka nova revizija je priča za sebe. Pred sobom imate paket koji je zaokružen u svakom pogledu. Svi alati su inicijalno tu, performanse podignute na izuzetan nivo, okruženje prilagođeno maksimalnoj udobnosti i efikasnosti programera, mnoge dopunske biblioteke postale su nepotrebne. Kao dugogodišnji korisnik *Clipper*-a ne mogu da sakrijem svoje oduševljenje. DOS će još dugo biti prisutan među nama, ovo je alat koji će ga pratiti u stopu i prerasti. Ako ste na početku rada sa bazama podataka, obavezno zastanite pred ovom alternativom - moćno, jednostavno. Ako ste korisnik neke od starijih verzija, pređite na 5.3, nećete se pokajati.

Autora možete kontaktirati na e-mail adresu dvesic@sezam.co.yu ■

Kako to radi OPO

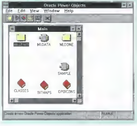
Uhvatili su me u trenutku kad sam prebacivao ikonu iz jednog prozora u drugi; mislili su da igram pasijans, a ja sam razvijao aplikaciju u *Oracle Power Objects*-u. Sreća je da sam brzo postigao rezultat; inače bih hleb možda već danas tražio negde drugde.

Milan Nikolić

Deo *Oracle Workgroup 2000* o čemu smo pisali u "Računarima" broj 108 je i *Oracle Power Objects* (*OPO*). Čemu i kome je *OPO* namenjen? *OPO* je dizajniran za okruženje radne grupe na mestima na kojima su otisci *Developer/2000* i *Designer/2000* preveliki: mesta poput *notebook* računara sa samo 4MB RAM-a, okoline koja je do sada bila prepuštena Access-u i *Visual Basic*-u.

Pored toga što je namenjen generisanju formi i izveštaja (*front-end* aplikacije) *OPO* može da radi sa internom ili eksternom bazom (*back-end* aplikacije). Interna baza, zvana *Blaze* baza, kreira se i održava od samog *OPO*-a. *Blaze* baza je podskup *Oracle7* baze i sadržana je u jednoj datoteci operativnog sistema. *Blaze* baza ne podržava procedure iz baze, kao ni bazne *trigger*-e, snabdevena je sekvencama kao i planiranom, ali još nedovršenom podrškom za tip podataka *long raw* ili *Binary Large Object* (BLOB). Ako Vam ponuđene funkcionalnosti ove ne mnogo zahtevne baze nisu dovoljne, *OPO* ima ugrađenu kontrolu za rad sa *Oracle7* serverom, kao i ugrađeni drajver za SQL server, a na putu je i podrška sporom ODBC-u (*Open Database Connectivity*).

Desktop OPO-a



Kako razviti jednostavnu aplikaciju

Instalacija *OPO*-a ne razlikuje se od bilo koje *Microsoft* instalacije. To je *push button* proces. Kada podignete *OPO*, prvo što vidite je desktop koji se sastoji od aplikativnih *folder*-a (folder je termin koji će zaživeti sa *Windows*-om 95, u slobodnom prevodu označava fioku) i baznih sesija. Aplikacioni folder sadrži sve elemente potrebne za izgradnju kompletne aplikacije (forme, izveštaje, klase i *bitmap*-e). Bazna sesija sadrži informacije o konekciji (na primer, ime, lozinku i mrežnu adresu servera na kojem je baza smeštena) i objektima baze kao što su tabele, pogledi, sekvence i indeksi. Kada aplikacija zahteva podatke od bazne sesije, sesija automatski šalje odgovarajući zahtev bazi i obrađuje rezultat. Kada se sesija "nakači" na bazu (uspostavi se konekcija), ona postaje "kontejner" za objekte baze koji su smešteni u toj bazi. Takva sesija naziva se aktivna i omogućava pregled sadržaja, modifikaciju, kreiranje i brisanje objekata baze. Aplikacija može istovremeno koristiti više baznih sesija.

Pre nego što se počne izgradnja aplikacije, mora se izvršiti konekcija na bazu. Mi smo se konektovali na tabele *Oracle* baze, koristeći, među korisnicima *Oracle*-a već čuvenog scott-a sa lozinkom tiger, čemu se pri svakoj instalaciji *Oracle* servera pridružuje nekoliko dobro osmišljenih tabela koje, između ostalog, sadrže informacije o odeljenjima i zaposlenima neke kompanije. Da bi se kreirala nova bazna sesija, iz **File** menija izabere se opcija **New Session**. Na ovakav niz akcija ispred Vas će se pojaviti "čarobnjak" ili kod nas već odomaćeni izraz *Wizard*. (*Wizard*-i su alati koji pojednostavljuju složene zadatke koristeći neke pretpostavke o procesu koji izvršavate. Kada se familijarizujete sa *OPO*-om, moći ćete da sve pretpostavke promenite i finije podesite kontrolu aplikacije.)

Prvo što ćete morati da odlučite u susretu sa *wizard*-om je koji tip baze želite. Premda smo izabrali da se konektujemo na *Oracle* i lažno predstavimo kao scott (što je kažnjivo po pozitivnim propisima, ali učinićemo sve za "Računare") izaberimo *Oracle* kao tip baze. Ukoliko koristite *Personal Oracle7*, dovoljno je da unesete samo ime i lozinku razdvojene kosom crtom: **scott/tiger**. Ako pak posedujete *client/server* verziju *OPO*-a i želite da se konektuje na neki server *Oracle7* baze, moraćete da ukucate kompletan string za konekciju. Na primer, konekcioni string za neku *UNIX* mašinu, pri čemu koristite *SQL*Net TCP/IP*, izgledao bi "**scott/tiger@T:nbj:oracle7**". Konekt string u svakom slučaju možete saznati i od svog administratora baze, ukoliko koristite *Oracle*.

Nakon što ste dali odgovore *Wizard*-u i kliknuli mišem na dugme **OK**, *OPO* će Vas upitati za ime sesije. Sesija se ne pravi samo za jednu aplikaciju, zato joj dodelite ime koje će Vas sećati na to šta ona sadrži. Ova naša biće imenovana po legendarnom scott-u. I u ovom dijalogu nađite dugme **OK** i pritisnite ga. Nakon što ste izvršili opisanu akciju, pojaviće se prozor u kojem se nalazi ikona sa imenom **SCOTT**. Dvostrukim pritiskom na nju, istim onim kojim startujete pasijans, Vi u stvari vršite konekciju na bazu. Ako sa bazom nema problema (na primer, zaboravili ste da je podignete) *OPO* će ispisati ikone koje pokazuju tabele, poglede, indekse i sekvence koje su Vam na raspolaganju.

Kreiranje forme

Kako su se objekti baze pojavili u prozoru sesije, iskoristimo ih za kreiranje jednostavne forme. No, prvo treba kreirati novu aplikaciju. Postupak nije ni za dlaku teži od onog

Aktivna bazna sesija



Toolbar

New Application New Session New Library Open existing Application Session Library Save Cut New Form New Report New Class Proprety sheet Generate executable aplication Run Application Run form report table

koji smo obavili pri kreiranju sesije. Iz **File** menija izaberite opciju **New Application**. *OPO* će Vas upitati za ime. S obzirom na to da se radi o zaposlenima, izaberimo ime EMPLOYEE. Pošto ste kreirali novu aplikaciju, došao je red da kreirate i prvu formu. Selektujte prozor sa imenom EMPLOYEE i iz *toolbar*-a izaberite ikonu **New Form**. *OPO* će Vam dati uokvireno "platno" na koje možete da razbacujete objekte iz *tool* palete, kao i listu u koju upisujete osobine svakog objekta (*property*-e).

OPO dozvoljava podešavanja formi ekrana poput *Visual Basic*-a, ali njegova glavna prednost sastoji se u tome da se forme mogu kreirati bez mnogo rada ili kodiranja. Ova prednost izvedena je korišćenjem *drag-and-drop* metoda: "zakačite" tabelu iz prozora bazne sesije i "ispustite" je na platno (ovakav postupak Vam je sigurno poznat, identičan je onome kada u pasijansu prebacujete kartu sa jedne "gomilice" na drugu). *OPO* će pretražiti bazu, smestiti sve kolone te tabele na ekran i postaviti svu neophodnu transakcionu logiku za baznu aplikaciju.

Kreirajmo, na primer, novu formu koja sadrži informacije o odeljenjima kompanije. "Zakačimo" tabelu DEPT (tabela odeljenja kompanije) iz prozora bazne sesije i dovucimo je u novu formu. *OPO* će smestiti labele (nazive) kolona i polja koristeći podrazumevani raspored koji zavisi od broja polja tabele. Pošto ste podesili raspored polja i labela po ukusu, dodajte *scroll bar* (bilo vertikalni ili horizontalni) koji će Vam omogućiti da prelazite sa sloga na slog pritiskom na vertikalni ili horizontalni *scroll bar* u paleti alata i kliktanjem na mesto na koje će se nalaziti gornji levi ugao *scroll bar*-a. Razvucite *scroll bar* po želji. Ono što ste postigli je jednostavna maska sa ugrađenom transakcionom logikom koja omogućava dodavanje novih, kao i ažuriranje i brisanje postojećih zapisa tabele DEPT. Forma je snabdevena i mogućnošću postavljanja upita po svim poljima. Ispitajte je pritiskom na ikonu *Run Form* iz *toolbar*-a.

Podešavanje aplikacije

Kao što ste se i sami uverili, kreiranje je vrlo jednostavno. Međutim, mala je korist ukoliko ne možete izvršiti potrebna podešavanja, kao što je promena imena forme. Da bi ste ovo postigli, kliknite *Static text* u paleti alata, a zatim na mesto na formi na kojem želite da se to ime nađe. Uredite text ukucavajući, na primer, "*Department*". Ako Vam se ne sviđa sistemski font, u formularu za osobine objekata (*Propreties sheet*) u stavci *FontName* izaberite neki od raspoloživih fontova pod *Windows*-om. Veličinu fonta promenite u stavci *FontSize*.

Ako su Vam potrebne samo neke kolone tabele DEPT, recimo DEPTNO, DNAME (šifra odeljenja, naziv) selektujte neželjeno polje (LOC) iz forme koristeći miša, a zatim pritisnite tipku **DEL**. Aranžirajte kolone i njima pridružene labele. Da biste promenili labelu neke kolone (niste zadovoljni postojećom) kliknite je mišem, napravite kratku stanku, a zatim opet kliknite. Labela će biti selektovana i možete otkucati željeni tekst na mesto postojećeg.

Jednostavna maska za unos tabela koje su u vezi jedan-prema-više

Svaka forma u aplikaciji sadrži samo jedan izvorni slog i na taj način je pridružena samo jednoj tabeli. Da bi se ograničenje prevazišlo, koriste se "umetnute" forme i *repeater*-i. "Umetnuta" forma (dobila je naziv po tome što se poziva iz druge forme) omogućava da se prikazuje jedan slog druge tabele. *repeater* (koji u mnogome podseća na tabele u *MS Word*-u) dozvoljava prikaz više slogova druge tabele. Ne samo da ovo dozvoljava izdavanje više tabela na istoj formi, već to omogućava da ta forma sadrži i tabele iz drugih baza.

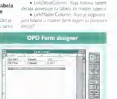
Prikazivati više tabela na istoj formi ima smisla ako su te tabele u nekoj vezi (postoji relacija između njih). Takve veze mogu biti jedan-prema-jedan (jednom zapisu jedne tabele odgovara jedan zapis druge tabele), jedan-prema-više i više-prema-više. Te veze mogu biti jake i slabe. Dve tabele koje su u vezi jedan-prema-jedan mogu se svesti na jednu tabelu, ali se ovakva veza često koristi u slučajevima kada je deo tabele slika ili sličan atribut tipa LONG RAW u *Oracle*-u ili tipa MEMO u Access-u. Na primer, ukoliko pravite formu za unos radnih sati zaposlenog, tom prilikom nećete prikazivati njegovu sliku, a upiti u tabeli koja je ne sadrži obuhvatiće mnogo manji rad sa diskom (ne učitavaju se velike kolone tabele) i na taj način se poboljšavaju performanse.

Najčešće korišćena veza u relacionim bazama podataka je jedan-prema-više. To je veza u kojoj jednom zapisu jedne tabele odgovara više zapisa druge tabele. Ovakva veza zove se i *master-detail* veza. Takvih primera u praksi ima mnogo. Na primer, odeljenja u nekoj organizaciji i zaposleni te organizacije. Odeljenje se sastoji od jednog ili više zaposlenih, svaki zaposleni pripada nekom odeljenju. Relacija više-prema-više se uvođenjem nove tabele koja sadrži samo veze svodi na dve relacije jedan-prema-više.

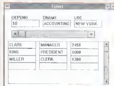
Da bi omogućio prikaz dve tabele u formi koje su u vezi jedan prema više (kraj jedan je master, a kraj više *detail*) *OPO* uvodi tri nove stavke formulara osobina: *LinkDetailColumn*, *LinkMasterColumn* i *LinkMasterForm*. Sve ove osobine definišu se na nivou detalja, što znači da nema nikakvih promena na nivou forme koja sadrži master da bi se izvršilo povezivanje sa detaljima. U setovanju veze pomoći će Vam odgovori na sledeća pitanja:

- *LinkDetailColumn*: Koja kolona tabele detalja povezuje tu tabelu za master tabelu?
- LinkMasterColumn: *Koja je odgovarajuća kolona u master formi kojom su povezani detalji?*

OPO Form designer



Jednostavna master-detail forma



- LinkMasterForm *Koja forma sadrži master slog za detalje?*

LinkDetailColumn i *LinkMasterColumn* ne moraju biti prikazani na formi.

Dodajmo formi koja već sadrži odeljenja *repeater* u kojem će se pojavljivati zaposleni iz tabele EMP. Selektujmo tabelu EMP iz bazne sesije SCOTT i startujmo je sa dva uzastopna kliktaja mišem. Dejstvo akcije koju ste upravo izveli je prikaz strukture tabele EMP u posebnom prozoru sa imenom EMP. Izaberimo neke od kolona (na primer ENAME, JOB, SAL) koristeći miša i **control** i/ili **shift** tipku – postupak je identičan onom u *FileManager*-u *Windows*-a. "Dovucimo" označene kolone u *repeater*. U formularu osobina popunite *LinkDetailColumn* sa **DEPTNO** (iz tabele DEPT), *LinkMasterColumn* sa **DEPTNO** (iz tabele EMP), *FormMasterForm* sa imenom forme. Jednostavna forma za unos *master-detail* veze je gotova.

Slike i sličice

Trenutno *OPO* se može koristiti pod *MS Windows*-u i na *Macintosh*-u. Obe platforme su grafičke, pa je i prirodno da u aplikaciju razvijenu u *OPO*-u možete dodati bitmapiranu grafiku. Dugmići (*button*-i) i forme (bilo standardne ili umetnute) su dozvoljena mesta. Kako aplikacioni folder sadrži sve što je potrebno aplikaciji, pre korišćenja sličica iste se moraju importovati u aplikaciju. Proces je podjednako lak kao i onaj za pravljenje novih foldera: iz **File** menija izaberite **Import BMP** opciju. Nakon što ste je uvukli u aplikaciju metodom *drag and drop*, možete je dodeljivati dugmićima i formama. Ako pak niste ljubitelj *drag and drop* metoda, u formularu osobina [illegible] stavku **BitMap** za imenovanje željene [illegible] **BitMap Tiled** ukoliko [illegible] da sliku ponov[illegible]

Izveštavanja

Lakoća izveštavanja (pravljenja izveštaja) je bitna pri izboru softvera. Pravljenje izveštaja u *OPO*-u je lako kao i pravljenje forme. Pošto ste selektovali aplikacioni folder [illegible] izaberite [illegible] *Report* [illegible] pred Vama će se pojaviti prozor za izveštaj koji je izdeljen na sledeće zone: *report header*, *page header*, *detail*, *page footer* i *report footer*. Iz aktivne bazne sesije koju smo imenovali SCOTT izaberite tabelu EMP i izaberite neke od kolona tabele (na primer EMPNO, ENAME, MGR, HIREDATE i SAL) postupkom koji smo već opisali, a zatim ih dovucite u zonu *detail*. Podesite fontove i veličinu koristeći formular osobina, uklonite okvire kolonama (*fontname*, *fontsize* i *HasBorder* stavke), preuredite labele na željeni način i izveštaj bi mogao biti gotov.

Ako zaposleni pripadaju nekom odeljenju, bilo bi dobro grupisati ih po pripadnosti istom. Iz palete alata (*tool palette*) izaberite ikonu *ReportGroup*, a zatim selektujte prozor izveštaja i kliknite na njega: pojaviće se još dve nove zone, *group header* i *group footer*, koje su pozicionirane pre i posle zone *detail*. Iz prozora u kojem je prikazana struktura tabele EMP izaberite kolonu DEPTNO (broj odeljenja) i, koristeći *drag and drop*, prebacite je u zonu *group header*. Izbrišite njenu labelu i podesite fontove. Očigledno, nedostaje nam ime tog odeljenja.

Iz palete alata izaberimo ikonu *Field* i označimo njeno mesto u zoni *group header*. U formularu osobina novog polja postavimo ime na **DeptName**, *DataType* na **String**, u *FontName* željeni font, *FontBold* i na **True**, a u *DataSource* unesimo:

=sqlLookup(scott, 'select dname from dept where deptno =" &deptno)

Selektujmo *group header* klikom na *Group Header* bar, u formularu osobina za njega u stavku *GroupCol* unesimo **DEPTNO**. Selektujte sve labele iz *detail* zone i prebacite ih u *group header* zonu, podesite ih da budu ispisane *bold* otiskom koristeći [illegible] opisani postupak. *Report header* iskoristite za unos naslova izveštaja, a *page header* za unos teksta koji će se pojavljivati na ostalim stranama izveštaja. Izveštaj je gotov. Pritisnite iz *toolbar*-a ikonu *Preview Report* i uverite se u lepotu sopstvenog dela.

OPO dozvoljava da definišete polja tipa datum štampe i broj strane koristeći funkcije **date()** i **pageNum()** u *DataSource* stavci, ispred kojih stavljate znak =.

Za kraj

Za razliku od drugih *Oracle* alata koji koriste *PL/SQL* jezik, *OPO* ima je *Oracle Basic*. *Oracle BASIC* kompatibilan je sa *Visual Basic*-om, pa će u razvoju aplikacija prednost imati VB programeri. Za kompletan razvoj aplikacije svakako će Vam biti potreban i jezik servera koji se razlikuje od *Visual Basic*-a.

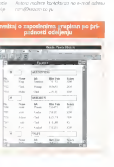
Aplikacija razvijena u *OPO*-u može imati OLE2 objekte, u nju se mogu ugrađivati razne OCX kontrole i iz nje se mogu pozivati funkcije iz DLL biblioteka, uključujući i MS SDK.

Oracle polaže velike nade u sve komponente *Oracle Workgroup 2000*, pa i u sam *OPO*, što pokazuje i agresivnim nastupom na tržištu. Od početka ove godine *Oracle* je podelio oko 30 000 CD-a sa kopijom *Oracle Workgroup 2000* koju možete testirati 90 dana.

Šta reći posle svega sem *So long, OPO*!

Autora možete kontaktirati na e-mail adresu nimi@sezam.co.yu

Izveštaj o zaposlenima grupisan po pripadnosti odeljenju



Isto to, samo malo drugačije

Od nastanka računara do danas nije bilo veće buke od one koju je izazvala pojava Objektno Orijentisanog Programiranja (OOP). Objekti su postali najvažnija stvar u programiranju, a programeri se ne mogu smatrati ozbiljnima ukoliko ne znaju bar jedan OOP jezik. Danas je šik biti objektno orijentisan - zašto se i vi ne bi pridružili klubu?

Miljan Mitrović

Pokušaćemo da u ovom kratkom tekstu objasnimo neke osnovne pojmove objektnog programiranja, čisto sa teorijske strane, ne vezujući se za određenu platformu ili alatku. Najveći problem vezan za većinu tekstova (ali i knjiga) ovog tipa je što sve zvuči veoma logično, toliko logično da pomislimo da o tome ne treba mnogo voditi računa. Preskočimo osnove i krenemo grlom u jagode. Praksa je, međutim, pokazala da se bez poznavanja nekih osnovnih pojmova, kao što su apstrakcija i polimorfizam, ne može uspešno shvatiti filozofija objektno orijentisanog programiranja, čime se veoma praktična i dobra tehnika programiranja pretvara u mučenje.

Njegovo veličanstvo, Objekat

Postoji više definicija objekta, sve zavisi od toga ko vam tu definiciju saopšti. Teoretičari bi rekli da je objekat neka celina, zamišljeni predmet koji kombinujete sa drugim predmetima preko njegove spoljašnosti. Sistemaš bi vam, sa druge strane, objašnjavao kako je objekat parče memorije koje se ponaša po nekim korisnički definisanim pravilima. "Fiziološki" posmatrano, objekat je spoj podataka i njima pripadajućih funkcija (koje se u ovom slučaju nazivaju metodama). Ukoliko ste ikada pisali neki program u kojem ste definisali neki svoj nestandardni tip (na primer kompleksni broj), sigurno ste pisali prateće funkcije za manipulaciju nad tim tipom. Jednostavno posmatrano, sve to možete strpati u jedan objekat. Na ovom nivou još nemamo nikakve koristi od OOP-a, pošto je celu stvar još uvek lakše uraditi strukturno. Ovde na snagu stupaju tri pojma vezana za OOP: apstrakcija, nasleđivanje i polimorfizam.

Moje, tvoje, naše

Apstrakcija, banalno prevedeno, predstavlja uopštavanje komplikovanih pojmova bez "gubljenja" suštine. "Tehnički" posmatrano, automobil ima motor sa unutrašnjim sagorevanjem koji, opet, ima cilindre, klipove, ventile itd. Apstraktno gledano, kolima se vozimo od tačke A do tačke B. Takođe, proizvođač koji koristi proizvedene sirovine ne mora da zna kako se one proizvode, niti koje su njihove sirovine da bi mogao da ih ugradi u svoje proizvode.

Kakve sad to ima veze sa programiranjem? Jednostavo rečeno, da bismo koristili gotov objekat, ne moramo da znamo kako je on pravljen niti šta je u njemu; sve što je nama potrebno je njegova veza sa spoljnim svetom (**interface**) kao i znanje o tome šta objekat radi. Takođe, kada mi pišemo objekat, apstrakcija nam služi da objekat učinimo stabilnijim i sigurnijim, kao i da zaštitimo neke naše ideje od krađe.

Kako to izvodimo? Jednostavno, kritične podatke i metode koji ne bi smeli da se pozivaju niti menjaju van objekta zaštitimo od pristupa. Postoje tri vrste podataka i metoda unutar objekta: *javne*, *zaštićene* i *privatne*. Javne podatke i metode može da menja ko hoće i oni uglavnom služe za to da se napravi veza objekta sa spoljnim svetom. Zaštićene metode i podatke "vidi" taj objekat i svi koji ga nasleđuju, dok privatnim podacima i metodama može da pristupi samo objekat unutar kojeg su definisane.

Dobro pisan objekat ima sve kritične delove deklarisane kao zaštićene ili privatne kako bi se sprečilo spoljno menjanje. Ukoliko na primer neki objekat prati broj poziva nekog metoda u nekoj promenljivoj, ta promenljiva bi trebalo da bude privatna. Ukoliko bi bila javna, promenljiva bi mogla biti promenjena, čime bi se objekat možda učinio nestabilnim.

U principu, treba ostaviti samo neophodan broj metoda kao javne, kako bi se sa objektom moglo "razgovarati". Sve podatke treba proglasiti zaštićenim ili privatnim, a upis i čitanje iz njih kontrolisati preko odgovarajućih javnih metoda. Ovako napisan objekat, ukoliko su javne metode dobro napisane, ima velike šanse da funkcioniše bez problema. Još jedna prednost apstrakcije je u manjem kodu. Za svaku javnu funkciju koja prima parametre gotovo uvek morate da pišete kod koji će kontrolisati opseg ulaznih vrednosti. Pošto zaštićene i privatne metode koristite samo vi, znate i opseg u kojem su parametri, pa su testovi nepotrebni.

Da bi apstrakcija imala svoju punu primenu, objekti moraju obuhvatati jednu celinu, da budu kompletni. Znači, ako imamo objekat kompleksnog broja koji ima metode za sabiranje i oduzimanje, nama je neupotrebljiv ako treba da pomnožimo dva kompleksna broja, jer opet sve moramo da uradimo mi, direktno pristupajući podacima za objekta (ukoliko su uopšte javni). Sa druge strane, treba napraviti kompromis između onoga što nama treba i onoga što se može napraviti. Ne isplati se pisati objekat sa sto metoda od kojih nama trebaju samo dve.

Nasleđivanje

Sada dolazimo do naziva ovog teksta: Isto to, samo malo drugačije. Nasleđivanje je prva osobina objektnog programiranja koju budući OO programeri savladaju. Ono je osnova razvoja objekata, sistem koji nam omogućuje da napravljene stvari ne pišemo ponovo (**code reusability**), već samo nadogradimo onime što nama treba, za trenutni specifični problem.

Jednostavno rečeno, bez nasleđivanja nećete nigde stići u svetu objektno orijentisanog programiranja, pošto će vam jedina preostala primena biti u pisanju gomile baznih objekata (objekti koji se nalaze na vrhu nasledne hijerarhije, tj. objekti koji ne nasleđuju druge objekte) koji će se tretirati kao klasični slogovi, uz dodatak automatske inicijalizacije polja. Iz ovoga treba da vam bude jasno da programi mogu biti objektni i "objektni". Od veštine samog programera zavisi koliko će program stvarno biti objektan.

Nasleđivanja ima raznih: jedan na jedan, više na jedan, sve zavisi od implementacije koju određeni jezik pruža. Cilj nasleđivanja je preuzimanje osobina jednog i više roditeljskih objekata i njihova dogradnja na potreban nivo, bilo dodavanjem novih metoda i podataka, bilo promenom već postojećih. Na primer, ako imamo već gotovu klasu kompleksnih brojeva koja sadrži metode za sabiranje i oduzimanje, nju možemo naslediti i dodati joj metode za množenje i deljenje. Takođe, možemo promeniti metod za sabiranje tako da, na primer, javi korisniku ako je rezultat van nekog opsega.

Kod primene *code reusable* tehnike najveći problemi su, zapravo, na samom početku rada, pošto je potrebno neko vreme

...da se napravi familija objekata koju biste kasnije koristili. Međutim, kako vreme protiče, doći ćete u situaciju da komlikovane programe možete napisati u roku od par dana, pošto ste većinu stvari već uradili za potrebe nekih drugih programa. U tom trenutku OOP prestaje da bude "maltretiranje" i "dodatno kucanje" i postaje zahvalna tehnika programiranja.

Klasa i instanca

Ova dva pojma su najzaslužnija za zbrku u OO jezicima. Klasa je samo definicija objekta, znači koje klase nasleđuje, koje metode se menjaju i/ili dodaju, koji podaci se dodaju. Ona, u stvari, predstavlja uputstvo za ponašanje objekta, i u praktičnom smislu predstavlja mrtvo slovo na ekranu. Da bi neka klasa "oživela" potrebno je da ima svoju instancu, tj. radnu kopiju u memoriji. Klasa je vezana za teorijski deo objektno orijentisanog programiranja, dok je instanca praktična primena. Jedno bez drugog ne mogu, pošto se instanca ponaša po opisima u klasi, dok je klasa neupotrebljiva bez svoje instance.

Jedna klasa može imati više instanci u memoriji, dok jedna instanca može pripadati samo jednoj klasi. Ne može instanca klase kompleksnog broja ujedno biti i instanca klase npr. matrica, dok u memoriji može postojati više instanci iste klase kompleksnih brojeva.

Ekvivalentan odnos ovome postoji u programiranju između tipova i promenljivih (više promenljivih, ali samo jedan tip, itd.). To je, ujedno, i razlog zbog kojeg se u većini OO jezika klasa tretira kao tip, a instanca kao promenljiva. Međutim, kao što ćete videti u narednim pasusima, klasna promenljiva i instanca koja se u njoj nalazi ne moraju biti iste klase, što je vrlo važno.

Polimorfizam

Kako definisati polimorfizam? Posmatrajmo na primer telefone - neki su crveni sa kružnim brojčanikom, neki su beli sa dugmićima, neki imaju automatsku sekretaricu, neki faks. Međutim, kako god oni izgledali, sve ih koristimo na isti način: podignemo slušalicu i izaberemo broj. I znate da ćete, bez obzira na to kako telefon izgledao, uvek moći da ga koristite. To je polimorfizam, neka vrsta standardizacije na nivou objekata: mogućnost da se n-ti objekat u naslednoj hijerarhiji poziva na isti način kao i prvi, drugi ili peti.

Polimorfizam je posledica jedne tehnike zvane kasno povezivanje (*late binding*), što bi otprilike predstavljalo linkovanje tokom izvršavanja. Tačna implementacija kasnog povezivanja varira od jezika do jezika, među tim zajedničko im je to da se u toku samog prevođenja ne zna da li će instanca nekog objekta biti baš te klase ili će biti instanca nekog izvedenog objekta. Zvuči komplikovano, ali nije uopšte. Primer: Imamo klasu **automobil** sa metodama koji ga opisuju i jednom specijalnom metodom koju ćemo nazvati **vozi** (očigledno, zar ne?). Tu klasu sada nasledimo i od nje napravimo klasu **kamion**. Promenimo metode koji opisuju objekat tako da on u stvari bude kamion (boja, broj točkova, nosivost itd.), ali metoda **vozi** ostaje ista, jer i kola i kamion imaju volan, gas, kočnice, kuplung i menjač. Voze se na isti način. Ukoliko bismo sad imali promenljivu **vozilo** tipa **automobil** i u njoj napravili instancu klase **kamion**, pri pozivu metoda **vozilo.vozi** ne bi bio pozvan metod klase automobil, već klase kamion.



Slika 1

Primene polimorfizma su neograničene. Pomenućemo samo jednu koja će vam najbolje dočarati njegove mogućnosti. Zamislimo neku igricu (iks-oks, pogodi broj itd.) i recimo da je igra dva igrača, bilo kompjuterska, bilo ljudska. Kreiraćemo tri klase (slika 1). Prva, bazna, nazvana **Igrač**, u kojoj je definisan metod **Igraj** koji ne radi ništa (takvi metodi, koji ne rade ništa i koji postoje samo da bi bili promenjeni u nekoj nižoj klasi, nazivaju se *apstraktni metodi*). Druge dve klase su **Komp** i **Čovek**, koje nasleđuju klasu **Igrač** i menjaju metodu **Igraj**, tako da **Komp.Igraj** izračunava potez koji bi kompjuter odigrao, a **Čovek.Igraj** traži od korisnika da unese potez. Takođe, kreiramo niz od dve promenljive tipa **Igrač**.

Verovali ili ne, ovime smo završili 99% igrice. Sve što sada treba da uradimo je da u taj niz ubacimo dve instance, bilo tipa **Komp**, bilo **Čovek**, i ciklično pozivamo metod **Igraj** tih instanci. Kako kaže Spike Lee, objekti će sami raditi pravu stvar. Iako je niz definisan kao dve instance tipa **Igrač**, kasno povezivanje će obezbediti da se ne poziva prazni, apstraktni metod **Igrač.Igraj**, već odgovarajući metod jedne od izvedenih klasa.

Mogli smo, umesto niza od dva **Igrača**, da napravimo niz od deset **Igrača**, to ništa ne bi menjalo u ostatku programa (klasama), samo u inicijalizaciji tog niza. Pogađate, igra bi radila isto tako dobro kao i sa dva igrača.

Umesto kraja

Svi ovi pojmovi objašnjeni su na teorijskom nivou, dok praktične primene zavise od jezika do jezika, a ovi opet imaju svoje "bubice" koje treba zadovoljiti, najčešće zbog tačnog funkcionisanja polimorfizma. Ne podržavaju svi jezici sve opcije OOP-a, na primer OO Pascal nema zaštićena područja, ali su nasleđivanje i polimorfizam svugde prisutni. U narednim tekstovima opisaćemo baš te implementacije, uz primer iks-oks igrice kao najjednostavnijeg programa koji se može pisati u OO stilu, a da obuhvata sve pomenute pojmove. Prošetaćemo kroz *OOPascal*, C++ i *SmallTalk*, jezik začetnik objektnog programiranja. ■

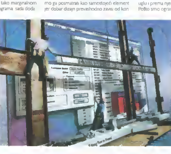
Autora možete kontaktirati na e-mail adrese mmitrovic@sezam.co.yu i mmix@galeb.etf.bg.ac.yu

Dizajnirajmo ekran

Nesumnjivo je da je *Microsoft Windows*-om uneo mnogo promena u računarski život. Neke od njih nam olakšavaju život, dok nam druge samo prazne džepove. Jedna svakako dobrodošla novina, posle godina DOS vladavine, je standardni grafički interfejs, odnosno izgled aplikacija. Pogledajmo kako stvari izgledaju sa programerske tačke gledišta

Ivan Sekulović

Sa naglom ekspanzijom *RAD* (*Rapid Application Development*) programiranja javlja se i sve veća mogućnost da programer upadne u zamku koju mu ono postavlja, a to je odlazak u kič. Dok su ranije loše dizajnirani programi nastajali uglavnom zbog lenjosti programera da se bakće tako marginalnom stvari kao što je izgled programa, sada dodavanje par nepotrebnih dugmadi, linija ili slika više ne znači pisanje desetina ili stotina linija koda, već samo nekoliko "klikova" mišem. Pošto je pisanje programa na ovim prostorima još uvek ponajviše *one man show* (ili je u boljem slučaju delo grupe programera), sve je prepušteno programeru, odnosno njegovom ukusu i osećaju za lepo. Šta to on treba da zna kada krene u razvoj neke aplikacije i koji su to postulati kojih bi morao da se pridržava?



Kako vidimo?

Gledanjem čovek spoznaje svet oko sebe. Ono što vidimo utiče na ono što osećamo. Gledamo očima, ali vidimo zahvaljujući mozgu. Mozak sabira deliće slike koje oko vidi i daje im značenje, prepoznaje ih. Iako niste svesni toga dok čitate ove redove, vi zapravo prepoznajete "oblike" slova, spajate ih u reči i povezujete sa nekim pojmom, odnosno značenjem, shodno vašem iskustvu. Čovek smatra tekst jednoličnim, dok na slike deluje mnogo emotivnije. Od nastanka izdavaštva ta ljudska osobina upotrebljava se u marketinške i propagandne svrhe. Često se novine kupuju zbog atraktivnih fotografija koje prikazuju neki aktuelan događaj. Za ono što se kaže jednom slikom bilo bi potrebno mnogo reči. Kada čovek posmatra ekran, on ga "pregleda" sleva na desno i od gore ka dole. Oko uvek pre spazi obojene elemente od crnih, izdvojene od onih u grupi i sliku od teksta.

Važno je zapamtiti da svaki deo ekrana kao celine utiče na ostale delove. Ne smemo ga posmatrati kao samostojeći element, jer dobar dizajn prevashodno zavisi od konteksta. Kod grafičkog korisničkog interfejsa (*GUI – Graphical User Interface*) izgled nekog elementa mora biti strogo povezan sa njegovom funkcijom. On mora da radi kao što izgleda, i da izgleda kao što radi.

Windows

Kako stvari stoje kod *Windows*-a? Nekoliko jednostavnih opštevažećih principa predstavlja osnovu *Windows* dizajna. Oni obuhvataju korišćenje i kombinovanje elemenata, boja, kontrasta i fontova.

Ako ste ikad radili u *Windows*-u (makar čistili "minice"), morali ste primetiti njihovu trodimenzionalnost. Ona se koristi za naglašavanje funkcija, kao i za davanje realnosti našim akcijama, tj. bolje rečeno, odgovorima na naše akcije. Sve što štrči može biti pritisnuto, a kada pritisnemo na njega, ulegne se, dajući našoj akciji fizički smisao, i to baš onaj koji odgovara kontroli na koju smo reagovali. 3D efekte postižemo uz pomoć senčenja. Zamišljeni svetlosni izvor u Windows-u nalazi se u gornjem levom uglu i prema njemu postavljamo sve senke. Pošto smo ograničeni sa 16 boja, odnosno dve nijanse sive (*dark gray, light gray*), za senčenje se koriste bela i crna boja (*white, black*).

Boje su u nama

Čovek je sposoban da razlikuje milione različitih boja, ali na svu sreću, sa bojama nije tako komplikovano kao što izgleda. Oko vidi samo tri boje: crvenu, zelenu i plavu, a sve ostale boje vidimo kao kombinaciju ove tri boje, što se može i matematički izraziti brojkama. Postoje tri osnovne karakteristike svake boje, i to su: boja (ime boje), zasićenost (intenzitet, jačina boje) i valer (gde se boja nalazi na skali od potpuno svetlog do potpuno tamnog).

Boje pomažu ljudima prilikom kontakta sa novim i dotad nepoznatim stvarima. Zašto? Ljudi vole boje, to je u čovekovoj prirodi. Ali, ne vole svi iste boje. Neko voli ljubičastu, dok je neko drugi mrzi. Stvar ukusa. Malo je poznato da je čak izreka *De gustibus non disputandum* (O ukusima ne vredi raspravljati) u stvari krnji oblik srednjevekovne skolastičke izreke koja je prvobitno glasila *De gustibus et de coloribus non disputandum* (O ukusima i bojama...). Dobro odabrana boja može uticati na to da korisnik program doživljava kao prijateljsko okruženje, dok nasuprot tome loše odabra-



... stvoriti cirkus od ekrana koji će ... Sećate li se filma *"Indiana Jones and ...* ... svetom gralu? Pravi daje večni ... je tako

Kao što smo već napomenuli, boja privlači oko, pa ju je dobro koristi za skretanje pažnje. Ali, oprez! Suviše boja može stvoriti konfuziju, kontekst je opet od presudnog značaja. Plava je loš izbor za sitan tekst ili druge male i tanke elemente, zato što je oku teško da se fokusira na nju, ali sa druge strane odlično pasuje za pozadine. Boje se mogu koristiti i za grupisanje srodnih elemenata. Sve, na primer, crvene objekte čovek smatra povezanim. Sa druge strane, čoveku povezivanje boje sa nekom funkcijom (plava za unos podataka, zelena za pregled i sl.) ne ide baš najbolje, pa nije preporučljivo koristiti samo boje u tu svrhu.

Pošto je izbor boja lična stvar, *Windows* to i prepušta krajnjim korisnicima tako što im omogućava da menjaju boju svakog *Windows* elementa (*Main, Control Panel, Color*). Naravno, preporučljivo je da se prilikom izbora boja uzimaju nenapadne boje, kao što su *default Windows* boje. Ukoliko se korisnik ipak odluči za *Hot Dog Stand Color Scheme* (a autor ovih redova se, na žalost, susretao sa takvim računarima), to je već njegov izbor, pa programer nema nikakvog razloga da lupa glavu oko toga.

Fontovi

"Slova k'o slova", reći će možda neko, ali nije baš tako. Fontovi imaju više funkcija od prostog obezbeđivanja slova za čitanje. Kao i ostali vizuelni elementi, fontovi utiču na stvaranje raspoloženja i motivacije. Pomoću njih se i stvara hijerarhija među informacijama - različitim veličinama fonta i masnoćom. Nisu ni sve vrste fontova podjednako pogodne za ispis na ekranu, *italic* i *Sans-Serif* fontovi su u principu teži za čitanje zbog svojih reckavih ivica, sve zbog premalog broja tačaka (piksela) za prikaz jednog karaktera. Tanja slova (debljine linije jedan piksel) i slova manjih veličina mogu predstavljati problem starijim osobama ili onima sa lošim vidom. Iako smo napomenuli da *Sans Serif* nisu baš najpogodniji fontovi, sistemski *Windows* font je baš jedna vrsta *Sans Serifa*, ali dorađena. Koristi se u veličinama od 10pt za meni, *title* i *status bar*, te 8pt za dijaloge. Mada bi to moralo da se podrazumeva, da ipak ne ostanemo nedorečeni - ukoliko program "priča" srpski, fontovi bez č, ć, š, ž i đ su neprihvatljiv izbor.

Dugmići

Kao što se dalo primetiti iz priče o trodimenzionalnosti *Windows*-a, glavno oruđe u komunikaciji predstavljaju dugmići (*buttons*). Oni bi grubo mogli da se podele na tri osnovne grupe: tekstualni (*text buttons*), grafički (*graphic buttons*) i upravljački (*control buttons*). Ove tri grupe imaju dosta toga zajedničkog - sličan izgled, isti vizuelni osećaj kada ih pritisnete, kao i slična stanja u kojima mogu da se nalaze (normalno, pritisnuto, isključeno i sl.). Kada odlučujete iz koje grupe da odaberete dugmiće za svoj program, morate se zapitati sledeće: gde ćete ga koristiti, šta će biti odziv na pritisak (neka funkcija, novi *form* ili nešto treće), kao i, naravno, koliko prostora imate.

Tekstualni dugmići obično se nalaze samo na nekoliko mesta u okviru aplikacije. Nisu previše lepi za oko, ali ukoliko smo tanki sa prostorom a upotrebljivost nam je na pr

[illegible] mogu biti sasvim zadovoljavaju[illegible] rešenje. Grafički dugmići su standardne veličine 24x22 tačke sa slikama od 16x15, a ukoliko nemamo dovoljno prostora možemo koristiti veličinu 22x18 sa slikama od 16x12. Pogodni su za korišćenje na *tool* bar ovima i *toolbox* ovima. Slike na njima moraju biti jednostavne i razumljive. One više treba da budu simboli nego slike. Ne treba da se zabeleži svaki detalj modela ili operacije, već je važno da se uoče karakteristike koje će ga jednoznačno predstavljati. Izgledom treba da budu približnije dečijim crtežima ili radovima majstora kubističkog slikarstva nego fotografijama. Ako je znak prepoznatljiv, bez poteškoća ga povezujemo sa procesom.

U slučaju grafičkih dugmića *hint* ovi bi trebalo da budu obavezan pratilac svakog dugmeta. Ako ste zaboravili, da vas podsetim, *hint* ovi su ona silna pojašnjenja od nekoliko reči koja se pojave ukoliko kursor zadržimo malo duže iznad nekog objekta. Ukoliko nam prostor to dozvoljava, može da bude interesantna i kombinacija teksta i slike na dugmetu. Iako nam mogućnosti dozvoljavaju druge kombinacije, tekst treba da bude ispod ili desno od slike, a sve ostalo je sasvim neprihvatljivo. Najzad, upravljački dugmići su u potpunosti standardizovani, pa bi ih bilo veoma teško zloupotrebiti.

Prostor za ovaj tekst je premali da bi se mogli upoznati sa svim principima i tehnikama dobrog dizajna, niti je to bila namera autora. Sve što je napisano su opšte poznate stvari i tekst treba više shvatiti kao podsetnik za ono što već znamo ili kao dodatni čas za one koje su u školi bežali sa likovnog. Autor će biti presrećan ako sledeći put, pre nego što stavite komponentu na *form*, malo razmislite kako će to sve izgledati, jer nije sve u algoritmima i optimizacijama koda. ■

Autora možete kontaktirati na e-mail adresu isekulovic@sezam.co.yu

Napoleonov princip

Biznis, industrija, finansijske institucije - od osnovnih do vrlo sofisticiranih usluga - ističu produktivnost kao sredstvo za povećanje profita. Dokazana produktivnost je postala imperativ današnjice

[illegible] dr *Milan Kukuka dipl. ing.*
[illegible] ing. *Ante Curlin*

Za razliku od proizvodnje čiji su se pro[illegible] godinama kontinuirano automatizovali [illegible] povećavali produktivnost, poslovni [illegible] su u značajnom zaostatku. Među[illegible] došlo je vreme kada su sakupljanje, obrada, smeštaj i distribucija informacija za [illegible] postali bitni isto koliko je bitna automatizacija proizvodnje.

[illegible] jedna mudra izreka koja je u po[illegible] svetu dobila status principa: Onaj [illegible] informacijama vlada i svetom. [illegible] se danas sa nekadašnjim istorijskim [illegible] sa zanimanjem otkrivamo [illegible] mnogi događaji [illegible] bitka za Britaniju [illegible] neobjašnjive pobede Romela [illegible] i niz drugih [illegible] onako kako su se [illegible] jer je pobednik znao informacije za [illegible] nije znao. I koliko god bila [illegible] dena kroz veličanstvene pobede [illegible] mih vojskovođa od Starog veka [illegible] dana, i koliko god se danas po[illegible] u elektronskom ratovanju, ova izreka [illegible] veću težinu u savremenom po[illegible] svetu. Razvijeni svet je to shvatio [illegible] zahvaljujući brzini prikupljanja, obrade [illegible] informacija doveo ostatak globusa [illegible] kolonijalan položaj.

[illegible] smo rešili da se ponašamo kao [illegible] poslovni svet, moramo da budemo [illegible] činjenice da je biznis, pre svega, [illegible] trka za informacijama. Shodno [illegible] suština funkcionisanja sistema upravljanja [illegible] kojima želimo da pričamo, trebalo [illegible] brzo i efikasno pronalaženje [illegible] potrebnih podataka i informacija. [illegible] važni resursi se u složenim poslovnim sistemima prikupljaju, definišu, selektuju, [illegible] razmenjuju, njima se upravlja, on se ko[illegible] kao i drugi resursi: radna snaga, mašine, materijal i novac. To je čuveni princip "5M" *Men, Machines, Material, Money & Messages*.

Ako pažljivije pogledamo ovaj princip, primetićemo da se čovek nalazi na prvom mestu. Čovek je kreator informacije i po današnjim najmodernijim teorijama o [illegible] vijanju ljudskim resursima osnovna poluga napretka. Firma koja danas ne ulaže u ljude i njihovo informatičko obrazovanje, firma kojoj se na čelu nalaze informatički neuki lju[illegible], osuđena je na propast. Jednostavno u podtekstu svega što je evidentna činjenica: sposobnost društva da se prilagođava [illegible] dernim svetskim tokovima [illegible] u broju kupljenih računara, već [illegible] učenom kadru koji će te računare i njihove mogućnosti znati maksimalno da iskoristi.

Ulaganje u informatičko opismenjavanje ljudi iz menadžmenta je osnovna pretpostavka uspeha. Danas se firme suočavaju sa velikim, često nerešivim problemima u [illegible] kovanju informacijama. Što vodi u svojevrstan paradoks: velike svote novca [illegible] se za računarsku opremu sa uverenjem da je time sve rešeno, a problem tek u počinju. Računarski [illegible] su nepotpune [illegible] [illegible] problem [illegible] narednih brojeva.

Dakle, informacija predstavlja osnovu za sve odluke. [illegible] zahteva pouzdanu, relevantnu [illegible] informaciju. [illegible] odluke [illegible] [illegible] luke [illegible] odgovaraju [illegible]

[illegible] (operativnog, taktičkog [illegible]) informacijama [illegible] stem [illegible] [illegible] sistema [illegible] [illegible] potrebne [illegible] Kao [illegible] je jednostavna [illegible] višenje [illegible] ta je [illegible]

[illegible] stvara u nepriliike [illegible] jednostavno [illegible] [illegible] [illegible] ove [illegible] ke p[illegible] [illegible] [illegible] strikt [illegible] što pod[illegible] strategije [illegible] tehnika [illegible] ■

Autor[illegible]
R[illegible]

Po protokolu

...lı kako to radı čuvenı DOOM pod mrežom Da lı možemo ı mı nešto slično primeniti ı u našim aplikacijama ? Naravno! Rešenje se krije u korišćenju nekog mrežnog protokola, u ovom slučaju to je IPX

Predrag Knežević

Zamislite sledeću situaciju pišete aplikaciju koja treba da radi na nekoj mreži a pri tome kopije koje se izvršavaju na radnim stanicama moraju međusobno da komuniciraju Problem se može rešiti na nekoliko načina Jedan koji prolazi kod svih mrežnih sistema je razmena poruka preko sistema za jednička fajlova (videti "U paukovoj mreži" Računari 110) Međutim ovo rešenje pored svoje univerzalnosti ima i nekih mana Prvo sami fajlovi koji igraju ulogu bafera moraju se nalaziti negde na serverovom disku Ukoliko zanemarimo pitanje njihove veličine ostaje problem lociranja tih fajlova od strane aplikacija. Dogovor mora postojati a on se obično sprovodi preko konfiguracionih fajlova Isto tako svaka radna stanica mora dovoljno često konsultovati svoj bafer kako bi što pre primila nove poruke Šta ako dođe do prepunjenja bafera ? Takođe ovakvo rešenje ne nudi mehanizam kojim bi sistem obaveštavao aplikacije o pristizanju novih poruka već vaša aplikacija mora o tome voditi računa

Posmatrajmo zato problem iz drugog ugla Kod svih tipova mreža protok informacija između stanica i servera odvija se na principu vremenskog multipleksiranja Svaka stanica samo u jednom kratkom vremenskom intervalu dobija dozvolu za korišćenje fizičkog medijuma kojim su povezane stanice Stoga se informacija koja se prosleđuje deli na manje delove pakete (**frames** u engleskoj terminologiji) i tako šalje kroz mrežu Svaki paket pored podataka koji se razmenjuju u sebi sadrži i adresu pošiljaoca adresu primaoca i nekoliko kontrolnih informacija. Svaka naša akcija na radnoj stanici (na primer otvaranje/zatvaranje/čitanje fajlova koji se nalaze na serverovom disku) propraćena je generisanjem niza paketa koji se prosleđuju serveru na obradu Po završenoj obradi server šalje stanicama pakete nastale kao rezultat obrade Samo generisanje paketa sakriveno je od nas programera Zato ne bi bilo loše da probamo da izbegnemo sve te više slojeve i da iz naše aplikacije formiramo pakete koji će putovati kroz mrežu Takav način rada iziskuje direktno opštenje sa slojem mrežnog protokola

Dobre strane ovog rešenja su veliko ubrzanje komunikacije jer se na taj način smanjuje broj paketa koji prolaze preko mreže pošto sada imamo punu kontrolu nad njima. Ovo nam može biti veoma bitno ako smo prinuđeni da često razmenjujemo veliku količinu podataka Dalje iz cele priče se izbacuje server ne postoji više potreba za postojanjem zajedničkih fajlova na disku pa keti se razmenjuju između samih stanica bez učešća servera

Mana je svakako sužavanje broja sistema na kojima se može izvršavati vaš softver jer njegov rad sada zavisi od postojanja upotrebljenog protokola S obzirom na to da je na našim prostorima od svih mrežnih sistema najzastupljeniji *Novell NetWare* u ovom tekstu ograničićemo se na komunikaciju preko njegovog **IPX** protokola

Checksum	2
Length	2
Transport control	1
Packet type	1
Destination network	4
Destination node	6
Destination socket	2
Source network	4
Source node	6
Source socket	2
Data	0-546

Paketi izbliza

IPX (*Internet Protocol Exchange*) je adaptacija **XNS** (*Xerox Network Standard*) protokola koji je razvijen od strane korporacije **Xerox** **IPX** podržava samo poruke datagram tipa tj ne postoji potvrda prijema od strane kojoj je poruka upućena Takođe ukoliko se zaredom pošalje više poruka ne garantuje se prijem u onom redosledu u kojem su i poslate S druge strane PC komunikacija poznaje i protokole *session* tipa kod kojih upućene poruke uvek stižu na odredište i to onim redosledom kojim su slate **SPX** je nadgradnja **IPX** a i predstavlja primer session protokola Zbog nedostatka kontrole kojom raspolaže **SPX** **IPX** je nešto brži a i mogućnosti primene su mu veće Iako se prijem poruka ne garantuje *Novell* tvrdi da 95% **IPX** paketa stiže na svoje odredište Ta brojka nam međutim ništa ne znači jer ćemo ipak u svojim rutinama morati uvesti potvrdu prijema

Svaki **IPX** paket sastoji se od zaglavlja i same poruke Struktura paketa pokazana je na slici 1 (fajl slika_1 pcx) Uočićemo karakteristična polja.

- **Checksum** - **Xerox** je ovo polje definisao u svom **XNS** protokolu, te se zbog kompatibilnosti pojavljuje u zaglavlju **IPX** paketa Niži protokoli koriste ovo polje da bi utvrdili da li je došlo do greške u prenosu a mi ga kao korisnici **IPX** a ne koristimo **IPX** ga uvek postavlja na FFFFH
- **Length** - veličina kompletnog **IPX** paketa (zaglavlje i podaci) Najmanja veličina paketa je 30 bajtova koliko je veliko samo zaglavlje dok maksimalna veličina iznosi 576 bajtova Veličinu ovog polja ne postavljamo direktno već **IPX** radi to za nas prilikom slanja paketa
- **Transport Control** **IPX** paket može da pređe preko 16 **NetWare** mostova (**bridges**) U trenutku slanja paketa **IPX** postavlja ovo polje na 0 a uvećava ga svaki put kada paket pređe preko mosta Kada brojač dostigne vrednost 16 paket se 'ubija'
- **Packet Type** **Xerox** je definisao različite tipove paketa koji se koriste u različite svrhe Za **IPX** pakete vrednost ovog polja mora biti 4 dok je za **SPX** pakete ova vrednost 5
- **Destination Network** identifikacija mreže (ili njenog dela) na kojoj se nalazi radna stanica koja treba da primi paket Ovaj broj zavisi od toga na kojem se segmentu mreže nalazi stanica različiti segment imaju različite brojeve i oni su fiksni za svaku mrežu administator ih dodeljuje pri instalaciji Ovo polje postavljamo mi kada želimo da pošaljemo paket
- **Destination Node** jedinstvena fizička adresa radne stanice Ovo šest bajtova dugo polje predstavlja serijski broj mrežne kartice a on je jedinstven za svaku karticu na svetu U ovo polje takođe upisujemo vrednost prilikom slanja paketa

• **Destination Socket** kompletna filozofija komunikacije preko **IPX** svodi se na rad preko soketa (*sockets*). Sokete (malo mi čudno zvuči prevod "utičnica") koji su preuzeti iz **UNIX** okruženja, otvaramo kada želimo da započnemo komunikaciju. Po uspostavljanju komunikacije pakete šaljemo preko otvorenog soketa, da bi ga na kraju zatvorili. Sokete možemo uporediti sa frekvencijama na skali radio prijemnika. Odabiranjem frekvencije počinjemo da primamo informacije od samo jedne stanice, ostale stanice kao da ne postoje. Slična stvar je i kod **IPX**-a. Otvaranjem soketa mi šaljemo i primamo pakete preko (uslovno rečeno) kanala, dok ostali paketi za nas ne postoje. Ovo postojanje soketa je samo logičko, uvodi ga protokol. Fizički, oni ne postoje, u jednom trenutku na kablu se može naći samo jedan paket. Postojanje soketa je vrlo značajno zbog višeprogramskih okolina, jer na taj način svaka aplikacija prima samo one pakete koji je zanimaju.

Ovo polje takođe postavljamo mi prilikom slanja paketa. Da bi stanica primila poslati paket, mora otvoriti soket čija je vrednost jednaka vrednosti polja **Destination Socket**. S obzirom na to da je ovo polje dvobajtna veličina, broj mogućih soketa je 65536. Naravno, neki od njih su već zauzeti od strane operativnog sistema, ali daleko veći broj je slobodan za upotrebu.

• **Source Network**, **Source Node** i **Source Socket** - imaju isto značenje kao i **Destination Network**, **Destination Node**, **Destionation Socket** respektivno, samo se odnose na radnu stanicu na kojoj se "vrti" naša aplikacija. Ove vrednosti postavlja sam protokol prilikom slanja paketa.

• **Data** - sami podaci koje želimo da pošaljemo.

Jedna važna napomena: sva višebajtna polja su u big-endian formatu, tj. viši bajt je na nižoj, a niži bajt na višoj adresi. Ovo je suprotno od prirodnog formata za PC računare, te o ovome moramo voditi računa prilikom popunjavanja polja. No, da pređemo na posao, da vidimo kako sve ovo izgleda u praksi.

Ima li koga kod kuće?

Ukoliko će se naš program "vrteti" isključivo na nekoj *Novell* mreži, onda možemo biti sigurni da postoji **IPX** protokol. Međutim, ukoliko to nije tako, potrebno nam je da utvrdimo da li na našoj mreži postoji predmetni protokol. Procedura je standardna, to činimo pozivanjem *Multiplex* interapta (2Fh), a pre toga popunjavamo relevantne registre (AX=7A00h). Ukoliko je **IPX** instaliran, u registru AL će se naći 0FFh, a kao bočni efekat u registarskom paru ES:DI pojaviće se ulazna adresa za **IPX**, čijim pozivanjem zahtevamo neku uslugu od protokola.

Skakanje na ovu adresu, koju dobijamo prilikom testa instalacije, nije jedini način da se zatraži neka usluga **IPX**-a. Drugi metod, koji je i stariji, koristi interapt 7A. Oba načina su potpuno ravnopravna, tako da je samom korisniku - programeru ostavljena mogućnost koji će od ova dva načina odabrati da koristi u svom programu. U svojim rutinama za komunikaciju koristio sam prvi metod, čisto radi neke dodatne sigurnosti. Naime, interapt 7A mogu koristiti i mnogi drugi na sistemu, tako da eventualno može doći do nekih "nesuglasica". Pozivanjem **IPX**-a preko ulazne adrese obezbeđujemo priličnu sigurnost našoj aplikaciji. Zahtevanje usluge od **IPX**-a prilično je standardno: u registar BX smešta se broj funkcije, u zavisnosti od zahtevane usluge popune se još neki registri i zatim se obavi skok na ulaznu adresu **IPX**-a. Posle završene obrade **IPX** u registru AL vraća kod greške (ukoliko je došlo do nje).

Pošto utvrdimo da **IPX** protokol postoji, sledeći korak u uspostavljanju komunikacije je otvaranje soketa. To činimo preko funkcije **IPXOpenSocket** (BX=00h), navodeći broj soketa i njegov tip. Broj soketa biramo sami, vodeći računa da naša aplikacija na svakoj radnoj stanici otvara isti soket. Dalje, možemo koristiti samo one sokete koje ne koristi sam sistem interno, jer inače može doći do nepredviđenih situacija. *Novell* je stoga opseg od 4000h do 8000h ostavio za dinamičku alokaciju i garantuje da ih ne koristi u interne svrhe. Sve ostalo može, ali i ne mora biti slobodno za upotrebu. Takođe samom sistemu možemo ostaviti mogućnost da on za nas odabere neki slobodan soket, ali to ima malo smisla, jer se unapred mora znati broj soketa, da bi komunikacija mogla da se uspostavi.

Što se tiče samog tipa soketa, razlikujemo kratkoživuće (*short lived*) i dugoživuće (*long lived*). Naime, prilikom završetka rada programa, sistem zatvara sve sokete koje je otvorila aplikacija, ukoliko to ona sama nije učinila. To je u redu, ali ne baš uvek: ukoliko pišete rezidentan program koji otvara sokete, prilikom pozivanja funkcije **Terminate&Stay Resident** **IPX** će i ovog puta zatvoriti sokete, iako to sada nismo želeli. Zato je uvedeno tipiziranje soketa: kratkoživući soketi ne preživljavaju izlazak iz programa, dok dugoživući, koji se ipak srazmerno malo koriste, to mogu.

Broj otvorenih soketa koji se mogu otvoriti sa jedne stanice je ograničen. Obično je podrazumevana vrednost 20, mada se **IPX** može podesiti tako da maksimalan broj bude 150.

Suprotna operacija od gore opisane je zatvaranje soketa. Izvodi se preko funkcije **IPXCloseSocket** (BX=01h) pre nego što naš program završi sa radom.

Event Control Block

Najveći deo ostalih usluga dobija se popunjavanjem strukture *Event Control Block* (**ECB**), koja se zatim prosleđuje **IPX**-u na obradu. **IPX** ih sve smešta u jedan, možemo uslovno reći, koš, a obrađuje ih tek kad nastupi događaj na koji se odnosi određeni **ECB**. Izgled **ECB**-a je pokazan na slici 2 (fajl slika_2.txt). Nama su od važnosti sledeća polja (mi ih postavljamo):

• **EventSvcRoutine** - adresa rutine koju će protokol pozvati pošto je **ECB** obrađen. Ukoliko se za adresu stavi 00000000h, skok se neće obaviti. Vrlo značajna mogućnost pomoću koje možemo realizovati baferisani ulaz/izlaz podataka koje šaljemo preko mreže. Prilikom pozivanja naše rutine od strane **IPX**-a važi sledeće:

Slika 2

```
ECB  =   Record
         LinkAddress                Pointer
         EventSvcRoutine            Pointer, {Event Service Routine}
         InUse                      Boolean,
         CompletionCode             Byte;
         SocketNumber               Byte2,
         ConnectionID               Word,
         Reserved1                  Word,
         Reserved2                  Array[1..12] of Byte
         ImmediateAddress           Byte6
         FragmentCount              Word,
         Fragments                  Array[1..2] of Record
         Address                    Pointer
         Length                     Word
End,
```

- interapti su onemogućeni
- vrši se daleki (FAR) poziv na našu rutinu
- svi registri su sačuvani
- u registarskom paru ES DI nalazi se adresa ECB-a koji je upravo obrađen
- DS registar ne mora pokazivati na naš DATA segment
- polje **InUse** u ECB-u sadrži vrednost nula (*true*)

Vršenje DOS poziva iz naše rutine nije sigurno pa se stoga ne preporučuje ali ako baš morate da ih izvodite onda to radite na isti način kao i kod drugih rezidentnih programa. Pozivanje većine **IPX** funkcija je dozvoljeno

- **SocketNumber** soket preko kojeg se vrši komunikacija isti onaj koji smo otvorili preko **IPXOpenSocket**. Potrebno je samo voditi računa da se on nalazi u *big-endian* formatu što je suprotno Intel-ovom predstavljanju višebajtnih veličina
- **ImmediateAddress** adresa radne stanice kojoj se šalje paket ili adresa bridža
- **FragmentCount** - broj delova koji će činiti paket. Obično je taj broj 2 (zaglavlje+podaci) mada taj broj može biti i drugačiji ukoliko su recimo podaci razbacani na više mesta u memoriji
- **FragmentAddress & Length** adrese i dužine pojedinih delova koji ulaze u sastav paketa ukupna dužina paketa ne može biti manja od 30 bajtova niti veća od 576
- **InUse** vrednost ovog polja ne postavljamo mi ali ga koristimo da bismo ustanovili da li je naš zahtev koji smo prosledili preko datog **ECB** a završen
- **CompletionCode** posle obrade datog zahteva uvidom u stanje ovog polja možemo utvrditi da li je došlo do neke greške ili ne (ukoliko je vrednost polja nula greške nije bilo)

Ostala polja u ECB-u **IPX** koristi interno. Ukoliko želimo da suspendujemo neki od zahteva koji smo poslali na obradu koristimo se funkcijom **IPXCancelEvent** (BX=06h ES DI=ECB). Ova funkcija ne poziva našu rutinu prilikom suspendovanja datog ECB a

Da bismo primili paket, ne treba nam više ništa od gore ispričanog. Najpre otvorimo odgovarajući soket popunimo ECB tj relevantna polja (**Socket Number**, **Event Service Routine**, **Fragment Count**, **Fragment Count & Address**) i pozovemo funkciju **IPXListenForPacket** (BX=04h ES DI=ECB). Proverom polja **InUse** utvrđujemo da li je paket primljen

Čiji si ti mali?

Da bismo poslali paket potrebno je još dosta posla. Najpre moramo odrediti adresu stanice kojoj želimo da pošaljemo paket. Za početak je potrebno da znamo samo korisničko ime korisnika koji je ulogovan na nekoj radnoj stanici. Adresu stanice određujemo pozivanjem standardnih *Novell* funkcija. Funkcija **GetObjectConnectionNumber** (AH=0E3h INT 21h) na osnovu korisničkog imena vraća listu brojeva. Članovi liste predstavljaju redne brojeve prijavljivanja stanice na mrežu (*connection number*). Ova lista može da primi najviše 100 elemenata. S obzirom na to da *Novell* ne zabranjuje korisniku da bude ulogovan na više stanica istovremeno potrebno je odrediti kojoj ćemo se stanici iz liste obratiti ukoliko je korisnik ulogovan na više njih. Obično uzimamo prvu stanicu sa spiska mada to ne mora biti pravilo

Kada konačno saznamo **Connection Number** stanice kojoj želimo da pošaljemo paket pozivamo funkciju **GetInternetAddress** (AH=0E3h INT 21h) koja nam na osnovu njega daje kompletnu adresu stanice na mreži (mrežni segment+adresa stanice). Iako sada već raspolažemo njenom adresom potrebno je još samo proveriti da li smo u mogućnosti da paket direktno pošaljemo na datu adresu ili moramo koristiti neki bridž. Pozivamo funkciju **IPXGetLocalTarget** (BX=02h) koja nam ili vraća adresu bridža ukoliko se on mora koristiti ili nam vraća istu onu adresu koju smo mu prosledili. Kao bočni efekat ova funkcija vraća i proračunato vreme za koje će paket stići na svoje odredište

Sada konačno raspolažemo svim relevantnim informacijama i možemo da popunimo zahtev za slanje paketa. U ECB u popunjavamo sledeća polja **SocketNumber**, **EventSvcRoutine**, **ImmediateAddress** vrednošću koju nam je vratila funkcija **IPXGetLocalTarget**, **FragmentCount** i **FragmentAddress&Length**. Dalje u **IPX** zaglavlju popunjavamo **PacketType** (dobija vrednost 4) **DestinationNetworkNumber**, **DestinationNetworkNode** i **DestinationNetworkSocket**. Na kraju zahtev prosleđujemo preko **IPXSendPacket** (BX=03h ES DI=ECB) i to je to

Postavlja se pitanje da li je moguće poslati paket svim stanicama u mreži (*broadcast*) a da pri tom ne moramo da određujemo adresu svake pojedinačne stanice? Rešenje postoji na nivou samog sistema takve pakete šaljemo na *broadcast* adresu (0FF FF FF FF FF FFh)

Cela gornja priča o određivanju adrese važi ukoliko je našoj aplikaciji značajno koji korisnik radi na stanici. A šta ako nam to nije od važnosti? Tipičan primer su igre kao što su **DOOM** i slične koje pri radu u mrežnom okruženju žele da znaju samo na kojim se sve stanicama izvršavaju a ne i ko radi na tim stanicama. Ovu informaciju nam ne može dati sistem već moramo malo pribeći programiranju

Ideja je sledeća: otvorićemo unapred dogovoreni soket i zatim ćemo poslati svim stanicama u mreži paket sa svojom adresom i određenim potpisom koji identifikuje aplikaciju. Istovremeno osluškujemo mrežu ukoliko se neka stanica javi na naš poziv. Ukoliko nam niko ne odgovori za neko vreme poziv ćemo ponoviti. Ceo postupak ponavljamo dogovoreni broj puta, pre nego što odustanemo. Stanica koja je primila zahtev za komunikaciju a i sama želi komunikaciju šalje stanici potvrdu zahteva zajedno sa svojom adresom. Istovremeno dodaje u svoju listu adresu stanice koja je uputila poziv. To isto čini i stanica koja dobije potvrdu prijema.

Na ovaj način moguće je saznati adrese proizvoljnog broja stanica na kojima se "vrti" naš program. Jedino što se ovde pojavljuje po prvi put jeste određivanje adrese naše radne stanice. To činimo preko funkcije **IPXGetInternetworkAddress** (BX=09h). Ono na šta se još mora obratiti pažnja jeste da naša mrežna kartica takođe reaguje na *broadcast* pakete koje je sama poslala pa ih zato moramo programski ignorisati. Možemo ih prepoznati tako što ćemo uporediti adresu pošiljaoca sa našom adresom

Slika 3

```
IPXHeader -  Record
        Checksum                 Word,
        Length                   Word,
        TransportControl         Byte;
        PacketType               Byte,
        DestNetworkNumber        Byte4,
        DestNetworkNode          Byte6,
        DestNetworkSocket        Byte2,
        SourceNetworkNumber            Byte4
        SourceNetworkNode        Byte6,
        SourceNetworkSocket            Byte2,
End,
```

Paketi u praksi

Na osnovu svega dosada iznesenog u ovom tekstu napisana je biblioteka za komunikaciju preko **IPX** protokola. Korišćenjem funkcija nekog nivoa napisane su rutine koje skrivaju od korisnika sve interne stvari, tako da korisniku - programeru ostaje samo da šalje svoje podatke. Pošto kod **IPX** protokola prijem nije zagarantovan, uvedena je i provera prijema paketa - redosled prijema je isti kao i redosled slanja i ne može doći do duplog primanja nekog paketa.

Saznavanje adresa stanica koje su raspoložene za komunikaciju vrši se preko funkcije **MakeLinks**, kojoj kao parametre prosleđujemo broj stanica sa kojima želimo komunikaciju, koliko puta ćemo probati da uspostavimo komunikaciju, kao i vreme koje ćemo na to utrošiti. Ukoliko se pojavi predviđen broj stanica u predviđenom vremenskom intervalu, kao rezultat dobijamo listu stanica sa njihovim adresama.

Ovu listu stanica dalje prosleđujemo funkciji **OpenComm**, koja preko zadatog soketa otvara komunikaciju sa stanicama. Treba napomenuti da je ulaz bafersan (veličina bafera je programibilna), dok izlaz to nije (u većini primena nije neophodno, mada se lako može dodati).

Slanje podataka obavlja se preko **PutMsg**. Poruka se šalje na sve stanice sa kojima komuniciramo, a ponavlja se sve dok sve stanice ne potvrde prijem. Sama rutina pored korisnikove poruke u sklopu samog paketa prenosi i podatke od internog značaja (detaljnije videti sam sors fajl NetComm.PAS). Uzimanje poruke iz bafera ide preko **GetMsg**. Ukoliko je bafer prazan, sačekaće se da pristigne prva poruka.

Kao ilustracija rada i korišćenja ove biblioteke napisana je i jedna mala demo igra. Stariji čitaoci se sigurno sećaju preteča današnjih SEGA aparata, koji su se pojavili u doba prvih Spektruma i Komodora. Reč je o tada popularnoj igri zvanoj TENIS. Svaki od igrača kontroliše jedan, uslovno rečeno, reket, a tu je i loptica koja se šeta po ekranu. Mada je ostavljena i mogućnost igranja u lokalu (dva igrača na jednoj mašini), daleko interesantnija je igra preko mreže. Kompletan izvorni kod ove igre, kao i same biblioteke za komunikaciju možete preuzeti sa Sezama (r:/rsoft/r113ipx.zip). ■

Autora možete kontaktirati na e-mail adrese pedjak@sezam.co.yu i pedjak@galeb.etf.bg.ac.yu

Literatura

Barry Nance: Network Programming in C

Prostori imena

U prošlom broju upoznali ste neke važne pojmove vezane za imena i njihovu deklaraciju. Sve to će vam već sada koristiti za razumevanje novog koncepta koji standard uvodi, a tekuće verzije većine prevodilaca ga još nemaju. Prostori imena omogućavaju lakše organizovanje i upravljanje velikim paketima klasa, što može znatno olakšati održavanje složenog programa.

Žarko Berberski

Stvaranje C++-a je, uvođenjem klasa i kontrole pristupa, donelo veliki napredak procesu razvoja programa. Prvih nekoliko godina masovne upotrebe ljudi su bili više nego zadovoljni onim što su dobili. Čitave biblioteke funkcija, sa gomilama izvornih spisa, prepakovane su u lako upotrebljive klase, broj grešaka se znatno smanjio, održavanje koda postalo je mnogo lakše - sve je izgledalo ružičasto. Velike firme se, međutim, nisu time zadovoljile. Želeći da što više iskoriste nove mogućnosti koje je doneo C++, programerski timovi počeli su da proizvode ogromne količine klasa, a zatim i velike biblioteke klasa. Za samo nekoliko godina, isti onaj haos koji je C++ jednom rešio, počeo je ponovo da se stvara, uključujući i probleme sa davanjem različitih imena svakoj klasi.

Zahvaljujući tome što je C++ bio još u sredini procesa standardizacije, rešenje za problem prevelikog broja klasa potražen je u dodavanju nove jezičke konstrukcije - **prostora imena**. Ruku na srce, najveći deo problema sa pretrpanim bibliotekama klasa je loš dizajn i nikakav programski jezik to ne može razrešiti, ali postoje softverski projekti gde je razgraničavanje prostora imena zaista potrebno i razumno rešenje.

Šta je zapravo prostor imena?

U C-u se prostorom imena nazivao skup imena određenog tipa unutar kojeg su sva imena morala biti različita. Postojanje četiri prostora imena (za oznake (*labels*) za skokove, za priveske (*tags*) struktura, unija i nabrojanih tipova, za članove strukture ili unije i za sva ostala imena) omogućavalo je delimično ponavljanje imena bez konflikta. U C++-u se takav pojam prostora imena praktično izgubio, tj. ostalo je samo to da oznake za skokove imaju svoj "prostor", dok je pitanje imena članova klase rešeno preko deklarativnog regiona.

Ako se prisetite, u prošlom nastavku ste videli da se najveći neprekidni deo programskog teksta u kojem ime može biti valjano naziva *deklarativnim regionom*. **Prostor imena je jednostavno eksplicitno zadat i ograđen deklarativni region, sa imenom ili bez njega.**

Tako gledano, čovek se mora zapitati nije li se sve to lepo moglo rešiti običnom klasom u kojoj su deklaracije svih klasa koje čine jedan paket. Na prvi pogled može, na drugi pogled - teško. Za paket od, recimo, 100 klasa od kojih deklaracija svake ima u proseku 50 linija, trebalo bi stvarati i održavati ogroman spis od 5000 linija (100 strana običnog teksta), i to bi sasvim uništio modularnost razvoja i održavanje programa, a o vremenu stalnog prevođenja ovakvog monstruma da i ne govorim.

Glavna razlika između ogromne klase i zadatog prostora imena je u tome kako se dodaju novi članovi. Kod klase se članovi moraju dodavati u zaglavlju u kojem je deklaracija klase i klasa se onda mora ponovo prevoditi, kao i sav kod koji zavisi od te klase. **Prostoru imena se novi članovi mogu dodavati bilo gde u programu i nikakvo novo prevođenje nije zbog toga potrebno.**

To je, međutim, tek početak. Deklaracija klase mora se nalaziti u jednom izvornom spisu. Obični deklarativni region, koji ste upoznali u prošlom nastavku, takođe se završava na granici izvornog spisa. **Prostor imena se može prostezati na više izvornih spisa.**

Prostor imena daje, dakle, veću fleksibilnost po pitanju prostiranja i dodavanja članova, no po pitanju kontrole pristupa daje mnogo manju fleksibilnost. **Svi članovi prostora imena su javni.** To je trenutno i jedina zamerka prostoru imena. Naime, kod velikog paketa javlja se potreba da neke klase budu javno dostupne, a druge ne, a da, pak, budu organizovane u prostor imena, kako bi se izbegli konflikti sa raznim bibliotekama. To se, naravno, može rešiti stvaranjem dva prostora imena, tako da samo za javni korisniku budu na raspolaganju spisi sa zaglavljima, no neki dizajneri velikih softverskih sistema žale se da to rešenje nije elegantno, da ne omogućava ostvarivanje dovoljno čistog dizajna programa. Drugi, pak, tvrde da je sve u redu, da oni prvi jednostavno treba da bolje promisle svoj dizajn, a ne da očekuju da programski jezik prati svaki njihov hir. Takve rasprave su, međutim, sport nekolicine ljudi koji zaista vode razvoj programa gigantskih razmera, a ovde su navedene samo kao ilustracija toga koliko je rad sa C++-om zapravo daleko otišao.

Zadavanje prostora imena

Za zadavanje prostora imena koristi se nova ključna reč **namespace**, za kojeg može (a i ne mora) biti ime, a telo definicije je, kao i kod svih drugih složenih konstrukcija, u vitičastim zagradama. Najbolje je da to pogledate na malom primeru:

```
namespace { int i; }
void f() { i++ }

namespace A
{  double c
```

```
}
namespace
{   int k,
    void f()
}
namespace A { i
```

u kojem su definisana dva prostora imena: jedan bez imena i jedan sa imenom A. Ovde takođe vidite kako definicija prostora imena ne mora biti neprekidna, već se uvek može dodati još neki član po potrebi.

Neimenovani prostor imena ograničen je na izvorni spis u kojem je definisan i služi kao pomoćno sredstvo, najčešće kao mora da se koristi biblioteka klasa ili funkcija koja stvara konflikte sa imenima koje već koristite u programu. Umesto da izmišljate nova, često sasvim nejasna imena, jednostavno konfliktni deo zatvorite u neimenovani prostor imena i sve je rešeno. Prevodilac će zapravo takvom prostoru dati neko privremeno ime, za koje se garantuje da je jedinstveno, tj. različito od bilo kojeg drugog u programu. Imena iz ovog prostora mogu koristiti kao obična imena, jer sve kućne poslove obavlja prevodilac. U prethodnom primeru će u definiciji prve funkcije

```
namespace { int i, } // posebno::i
void f() { i++ }     // posebno::i++
```

prevodilac ubaciti, kao što se vidi u komentarima, umesto reči **posebno** privremeno jedinstveno ime za prostor imena, ali programer koristi ime iz prostora kao da je obično, dakle bez posebnog kvalifikatora.

Za imenovani prostor imena je stvar drugačija. Njegovi članovi se moraju eksplicitno kvalifikovati imenom prostora (kao da je ime klase):

```
d = A::c * 31 517,
k = A::i - i,     // A::i     poseb-
no::i
e = A::f(7)
```

i u drugom redu vidite i razlog: u istom izrazu korišćene su dve promenljive istog imena, ali je svaka iz drugog prostora imena. Kad se članovi iz nekog prostora imena mnogo koriste, mogu se sva imena uvesti u tekući doseg pomoću ključne reči **using**. Istom naredbom može se uvesti i samo jedno ime. Od mesta uvođenja ime se koristi kao obično ime, ali podleže svim pravilima za obično ime, pa će se prevodilac buniti ako u tekućem dosegu nađe duplirano ime.

```
using A::c,
void a()
{   i++,                  // u redu
je    posebno::i
    e = f(7)      // greska - A::f
nije u dosegu
    c += 3 14     // u redu
}

using namespace A,
void a()
{   i++     // greska   posebno::i
...
    i(7)    // u redu je
    e = f(7)   // u redu je - A::f
    c += 3 14  // u redu je - A::c
}
```

Možete se podsetiti iz prošlog nastavka da ime deklarisano u imenovanom ili neimenovanom prostoru imena ima za doseg taj prostor imena, a da njegov mogući doseg uključuje i mogući doseg svake **using** naredbe koja navodi dati prostor imena.

Pošto je definicija prostora imena obična deklaracija, a u svom telu takođe sadrži deklaracije, to se prostori imena mogu gnezditi, mada tako nešto obično spada u nepotrebnu egzibiciju.

Deklaracija i upotreba članova prostora imena u mnogome je slična radu sa klasama. Kad i kod klasa, definicija prostora imena ne mora sadržati i definiciju svakog svog člana – samo deklaracija je dovoljna. Definicija se može kasnije zadati eksplicitno, ali samo u prostoru u kojem se nalazi i prostor imena kojem pripada.

```
namespace A
{   namespace B
    {
        int f(),
    }
    int V::f()    { /*     */ }
// u redu je
    int V::g()    { /*     */ }
// greska

// g() jos nije clanica
    namespace B
    {
        int g());
    }
}
namespace C
{
    ing A::B::g(){ /*     */ }
// greska
// A se ne nalazi u C
}
```

Svako ime deklarisano u prostoru imena pripada tom prostoru, čak i funkcije prijateljice klase koja se nalazi u prostoru imena. Pogledajte to na malom primeru:

```
namespace A
{
    clas X
    {
        friend void f(X); // dekla-
racija f-a
        class Y
        {
            void g(),
        },
    }-
    void f(X a) { /*     */ }
// definicija
    X x,
    void g()      { f(x); }
// f i g su clanovi A
}
```

```
using ...
void f()
{
    ... f(x)
    ... X::f(x),   // greska
    ...ca A::X
    ...X::Y::g()   // greska
...clanica A::X::Y
}
```

Alias za prostor imena

Kad se definiše prostor imena, obično se daje ime koje jasno opisuje njegovu namenu. To ime je često dugačko i zamorno za stalno ponavljanje. Umesto njega može se definisati kraće ime koje služi kao alias:

```
namespace vrlo_dugacko_ime_prostora
{
    float pi;
}
namespace VDIP vrlo_dugacko_ime_pro-
stora,
```

i koje se lakše može koristiti. Alias može zameniti originalno ime prostora imena u svim slučajevima korišćenja imena koja se nalaze u datom prostoru imena.

Dodavanje novih imena zahteva, pak, originalno ime prostora:

```
namespace VDIP
{
    int i         // greska - alias
}
namespace vrlo_dugacko_ime_prostora
{
    int i         // sve je u redu
}
```

Neki dizajneri objektno orijentisanog softvera koriste aliase za malo dodatno obezbeđenje od zabune, naročito za biblioteke klasa u kojima je poželjno da korisnik ne dodaje imena, već da samo koristi postojeća. Pošto jezik ne daje mehanizam "zatvaranja" prostora imena, u dokumentaciji se navede da korisnik treba da koristi samo alias, jer će ga tada prevodilac upozoriti kada je pogrešio.

Zatvaranje prostora imena je u direktnoj suprotnosti sa njegovom fleksibilnom proširivošću. Pretpostavimo da se doda naredba za "zatvaranje" protora imena, tako da posle nje prevodilac odbije da dodaje nova imena u zadati prostor imena. Dok prevodi jedan spis, prevodilac nikako ne može da zna da li je prostor "zatvoren" u nekom drugom, što takav zahtev čini nerealnim.

Korišćenje using naredbe

Korišćenjem **using** naredbe vrši se deklaracija, dakle uvođenje imena, ali ne nekog novog imena, već onog koje već postoji u nekom prostoru imena, pa se može reći da se vrši "uživanje" jednog prostora imena (ili njegovog dela) u drugi. To je u principu jednostavna stvar, ali postoji nekoliko detalja koje je dobro pogledati kao posebne slučajeve.

...me koje je "uliveno" može se kasnije koristiti kao svako drugo ime iz prostora imena u koji je uliveno.

```
void f();
namespace A
{
    void g();
}
namespace X {
    using ::f;      // globalna f
    using A::g;     // g je iz A
}
void h()
{
    X::f(); // poziva ::f kao da je iz X-a
    X::g(); // poziva A::g kao da je iz X-a
}
```

Značenje imena koje se uvodi **using** deklaracijom određuje se na mestu deklaracije, tako da imena koja se kasnije dodaju prostoru imena ne utiču na upotrebu ranije "ulivenog" imena.

```
namespace A
{
    void f(int);
}
using A::f;         // f je sinonim za A::f(int)

namespace A
{
    void f(char);
}

void foo()
{
    f('a');         // poziva f(int)
}                   // iako postoji f(char)

void bar()
{
    using A::f; // f je sinonim za i za
    // A::f(int) i za A::f(char)
    f('a');         // poziva f(char)
}
```

Prostori imena, u delu koji se odnosi na korišćenje članova u programu, slični su klasama, pa se naredba **using** u nekim slučajevima može koristiti i za klase. Korišćenje **using** naredbe za klase dozvoljeno je samo za imena u klasi koja se definiše i imena iz bazne klase, no i tu treba biti vrlo pažljiv jer to ima smisla samo u posebnim slučajevima, obično radi izbegavanja skrivanja imena kod izvedenih klasa. Funkcija članica izvedene klase će "na ime" sakriti istoimenu funkciju bazne klase (sem ako je ona virtualna), umesto da je samo preklopi. Pre uvođenja naredbe **using** morala se ponavljati definicija uz pozivanje funkcije iz bazne klase. Sada je to mnogo jednostavnije.

```
struct B
{
    void f(char);
    void g(char);
};
struct D : B
{
    using B::f;     // B::f(char) je uvedena
    void f(int);
    void g(int);    // B::g(char) je sakrivena
};
D d;
d.f(1);     // poziva D::f(int)
d.f('a');   // poziva B::f(char)
d.g(1);     // poziva D::g(int)
d.g('a');   // greska - B::g(char) je sakrivena
```

Primetite da se navodi samo ime funkcije. Ono se odnosi na sve preklopljene funkcije datog imena, a zadavanje nove funkcije u izvedenoj klasi sakriće samo odgovarajuću funkciju (sa istim tipovima parametara) i neće izazvati konflikt. Uvođenje imena virtualne funkcije neće ništa promeniti, ali neće ni javiti grešku.

```
struct B
{
    virtual void f(int);
    virtual void f(char);
    void g(int);
    void h(int);
};

struct D : B
{
    using B::f;     // zapravo ne radi nista
    void f(int);    // nova verzija B::f(int)
    using B::g;     // sprecava skrivanje
    void g(char);   // preklapanje
    using B::h;
    void h(int);    // D::h(int) skriva B::h(int);
            // nema konflikta imena
};
void k(D* p)
{
    p->f(1);        // poziva D::f(int)
    p->f('a');      // poziva B::f(char)
    p->g(1);        // poziva B::g(int)
    p->g('a');      // poziva D::g(char)
    p->h(1);                // poziva D::g(int)
}
```

Ime uvedeno **using** deklaracijom samo je alias originalnog imena, tako da ne menja tip, povezivanje i druge atribute originala. Pravila o kontroli pristupa za original takođe važe, pa je uvođenje aliasa privatnog člana bazne klase greška, baš kao da ste pokušali da mu direktno pristupite.

Naredba **using**, primenjena na kompletan prostor imena, je tranzitivna, što znači da i prostor imena koji se uključuje može u sebi imati **using** naredbu, da će sva tako uključena imena zajedno biti "ulivena" u novi prostor imena. To nekada može biti korisno, ali ako se ne planira dovoljno precizno, može doneti i dosta neočekivanih problema.

```
namespace M
{
    int i;
}
namespace N
{
    int i;
    using namespace M;
}
void f()
{
    N::i = 7; // u redu je - M::i nije clan N
    using namespace N;
    i = 7; // greska - konflikt M::i i N::i
}
```

Kod "ulivanja" celih prostora imena, sva imena dodata do mesta upotrebe nekog imena su aktivna, a ne samo ona koja su dodata do **using** naredbe (dakle, sasvim različito od "ulivanja" pojedinačnog imena). To takođe može dovesti do neočekivanih problema za onoga ko nije pažljivo planirao.

```
namespace D
{
    int d1;
    void f(char);
}

using namespace D;
int d1; // nema konflikta do pokusaja upotrebe

namespace E
{
    int e;
    void f(int);
}
namespace D
{   // dodavanje imena posle using-a
    int d2;
    using namespace E;
    void f(int);
}

void f()
{
    d1++; // greska - ::d1 ili D::d1?
    ::d1++;         // eksplicitno razresen doseg
    D::d1++;        // eksplicitno razresen doseg
    d2++; // D::d2
    e++;            // E::e
    f(1); // greska - D::f(int) ili E::f(int)?
    f('a');         // D::f(char)
}
```


Žarko Berberski je nezavisni konsultant za UNIX i hipermediju, C/C++ programer i tehnički autor iz Beograda. Autor je knjige "Programiranje na ANSI C-u" (Mikro knjiga, Beograd). Možete ga kontaktirati na zarko@galeb.etf.bg.ac.yu ili zarko@osmeh.fon.bg.ac.yu

Programirani DMA

Na samoj matičnoj ploči svakog PC računara, pored procesora i memorije, nalazi se i dosta prateće logike - kontroler tastature 8041, kontroler prekida 8259, tajmer 8253, DMA kontroler 8237. Ovog puta pozabavićemo se DMA kontrolerom.

Predrag Kneževic

Sama skraćenica **DMA** potiče od *Direct Memory Access*. Kao što mu samo ime kaže, **DMA** kontroler može direktno da pristupa memoriji bez posredstva procesora. Čemu to može da posluži već se može naslutiti. U većini programa veliki deo procesorskog vremena ode na opštenje sa periferijama: čekanje na podatke ili na njihovo slanje. Ukoliko se radi još o prenošenju velike količine podataka, onda procesor vrlo mali deo vremena radi nešto "korisno". Evo kako u suštini izgleda primanje podataka koje šalje neka periferija:

```
MOV AX,SEGMENT          ;segment odredišne adrese
MOV DS,AX
MOV DI,OFFSET           ;ofset odredisne adrese
MOV CX,COUNT            ;koliko bajtova
MOV DX,IOPORT           ;adresa periferije
READ  IN AL,DX          ;u petlji citamo podatke
MOV [DI],al             ;i smestamo ih u memoriju
INC DI
LOOP READ
```

Slične rutine bile bi i u slučaju slanja podataka ka periferiji. Na ovom mestu možemo uočiti nekoliko stvari. Prvo, u toku ovakvih operacija procesor je 100% zauzet i ne može da izvršava nijednu drugu stvar. Dalje, sam proces primanja i slanja podataka nije baš toliko jednostavan kao što bi se moglo zaključiti iz gornjeg koda. Recimo, sama periferija radi na nižem taktu od procesora, pa je samim tim i sporija i ne može da prima i šalje podatke onim ritmom koji bi odgovarao samom procesoru. Zato procesor dodatno vreme gubi na razna čekanja i sinhronizacije i kao konačan rezultat javlja se pad performansi. Bilo bi dobro kada bi ovaj prilično dosadan i jednostavan posao prepustili nekom malo inteligentnijem kontroleru, a procesor ce za to vreme raditi nešto korisnije. Rešenje je **DMA** kontroler.

Prema teoriji...

Kao i svaki kontroler, **DMA** kontroler vezan je na sve tri magistrale (adresnu, kontrolnu i magistralu za podatke). Obični kontroleri (kontroler tastature, interapt kontroler i sl.) mogu da čitaju podatke sa obe magistrale, ali su u stanju samo da generišu podatke koji izlaze na magistralu za podatke. Za razliku od njih, **DMA** kontroler može da generiše podatke koji izlazne na adresnu magistralu i par kontrolnih signala (da li se vrši obracanje memoriji, čitanje/pisanje). Kako iskoristiti prisustvo **DMA** kontrolera? To je moguće samo ako je periferija spremna da "sarađuje", tj. može da generiše potrebne signale.

Objasnićemo kako u principu teče prenos podataka iz memorije i u nju posredstvom **DMA** kontrolera. Prvi korak je generisanje signala **DMA Request** (**DREQ**) od strane same periferije. Ovime sam uređaj signalizira da treba da počne prenos u memoriji i iz nje. Način određivanja operacije koja se vrši (čitanje ili pisanje) obično se vrši preko još par kontrolnih linija i mnogo zavisi od praktične realizacije.

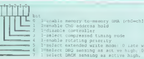
Pošto je procesor "gazda" u sistemu i u njegovom vlasništvu se nalaze obe magistrale, linije čitanje/pisanje po memoriji (**MEMR & MEMW**) i na uređajima (**IOR & IOW**), potrebno je najpre dobiti dozvolu za pristup istima. Pošto je **DMA** kontroler primio signal **DREQ**, on ka procesoru generiše signal **HOLD REQUEST**, čime ga obaveštava da mu što pre omogući izlaz na magistrale. Pošto je procesor kompletirao izvršenje tekuće instrukcije i pošto mu magistrale više ne trebaju, on ka **DMA** kontroleru generiše signal **HOLD ACKNOWLEDGE** i zatvara svoja tristatička kola preko kojih je priključen na adresnu i kontrolnu magistralu i magistralu za podatke.

Po prijemu signala **DMA** kontroler kreće na posao. Otvara svoja tristatička kola ka magistralama i daje periferiji signal **DMA ACKNOWLEDGE** (**DACK**). Ukoliko treba da se izvrši prenos podataka iz memorije, tada kontro-

Slika 1

```
0000   w     DMA channel 0 base address
       r         channel 0 current address
0001   w     DMA channel 0 word count
       r         channel 0 current word count
0002   w     DMA channel 1 base address
       r         channel 1 current address
0003   w     DMA channel 1 word count
       r         channel 1 current word count
0004   w     DMA channel 2 base address
       r         channel 2 current address
0005   w     DMA channel 2 word count
       r         channel 2 current word count
0006   w     DMA channel 3 base address
       r         channel 3 current address
0007   w     DMA channel 3 word count
       r         channel 3 current word count
0008   r     DMA status register
0008   w     DMA command register
0009   w     DMA request register
000A   w     DMA single mask bit register
000B   w     DMA mode register
000C   w     DMA clear byte pointer flip-flop
000D   w     DMA master clear
000E   w     DMA clear mask register
000F   w     DMA write mask register
0081   r/w   DMA channel 2 page register
0082   r/w   DMA channel 3 page register
0083   r/w   DMA channel 1 page register
0087   r/w   DMA channel 0 page register
0089   r/w   DMA channel 6 page register
008A   r/w   DMA channel 7 page register
008B   r/w   DMA channel 5 page register
00C0   w     DMA channel 4 base address
       r         channel 4 current address
00C2   w     DMA channel 4 word count
       r         channel 4 current word count
00C4   w     DMA channel 5 base address
       r         channel 5 current address
00C6   w     DMA channel 5 word count
       r         channel 5 current word count
00C8   w     DMA channel 6 base address
       r         channel 6 current address
00CA   w     DMA channel 6 word count
       r         channel 6 current word count
00CC   w     DMA channel 7 base address
       r         channel 7 current address
00CE   w     DMA channel 7 word count
       r         channel 7 current word count
00D0   r     DMA status register
00D0   w     DMA command register
00D2   w     DMA request register
00D4   w     DMA single mask bit register
00D6   w     DMA mode register
00D8   w     DMA clear byte pointer flip flop
00DA   w     DMA master clear
00DC   w     DMA clear mask register
00DE   w     DMA write mask register
```

Slika 2: DMA command register



ler generiše signale **IOW** i **MEMR**, a u slučaju prenosa u memoriju generišu se signali **IOR**, **MEMW**. Periferija se odaziva na signale **IOR** i **IOW** i prima/šalje podatke. Uz prisustvo signala **MEMW** i **MEMR** i adresa generisanih od strane kontrolera dolazi do upisa/čitanja podataka u memoriju. Po završetku prenosa kontroler ukida signal **DACK**, zatim i **HOLD REQUEST**, zatvara svoja tristatička kola i priprema se za sledeći transfer. Ukidanjem signala **HOLD REQUEST** procesor ukida svoj signal **HOLD ACKNOWLEDGE** i ponovo uspostavlja kontrolu nad magistralama. Sada kontroler generisanjem interapta ili preko statusnog registra signalizira da je prenos završen.

Posredstvom **DMA** kontrolera moguće je izvršiti prenos velike količine podataka odjednom. Postavlja se pitanje vlasništva nad magistralama za sve vreme prenosa. Postoje dva moda u kojima se može vršiti prenos.

BURST ili *BLOCK* mod - magistrala je u vlasništvu **DMA** kontrolera sve dok se ne završi prenos. Ovaj način rada pogodan je samo ako radimo sa nekom brzom periferijom, inače bi došlo do nepotrebnog čekanja od strane procesora, jer sve dok su magistrale u vlasništvu **DMA** kontrolera, procesor ne može da dovuče sledeću instrukciju iz memorije.

SINGLE mod - **DMA** kontroler posle svakog prenetog bajta/reči vraća magistrale procesoru i ponovo čeka priliku da je dobije. Ovakav način rada pogodan je za sporije periferije i u praksi se uglavnom koristi samo on, jer su sve periferije praktično dosta sporije od samog procesora.

Samo programiranje **DMA** kontrolera vrši se preko odgovarajućih registara koji su dostupni programeru. Slanjem odgovarajućih vrednosti vrši se inicijalizacija, postavljanje adresa, biranje moda rada i slično. Iako smo do sada uglavnom pričali o prenosu podataka od periferije i ka njoj, treba istaći da većina **DMA** kontrolera ima mogućnost prenosa podataka na relaciji memorija - memorija. Na žalost, u okviru same PC arhitekture ova mogućnost se ne koristi. Uopšte **DMA** kontroler se vrlo slabo koristi, kao posledica održavanja kompatibilnosti. Naime, sam kontroler, kao i magistrala, rade na daleko nižem taktu od procesora, tako da bi bilo apsurdno da radimo prenos podataka sa diska preko **DMA** kontrolera, ako je to sporije u odnosu na situaciju kada istu tu stvar radi procesor. Jedino gde možemo iskoristi ti prednost **DMA** kontrolera je u kontrolisanju zvučnih kartica, koje ne zahtevaju velike brzine prenosa, a na taj način procesoru dajemo mogućnost da radi neke druge poslove (osvežavanje ekrana, na primer).

A u praksi...

Svaka matična ploča, počevši od XT-a, ima na sebi, kao što smo već ranije rekli, **DMA** kontroler 8237. Kontroler 8237 može da sarađuje sa više periferija (uređaja) odjednom i svakom od njih posvećen je poseban kanal (*CHANNEL*). Shodno tome, postoje i četiri **DACK** linije označene sa **DACK0-3**.

- Prioritet ovih linija može se programirati preko komandnog registra **DMA** kontrolera, a postoje dva moguća načina:
- fiksan prioritet, pri čemu je kanal 0 najvišeg, a kanal 3 najnižeg prioriteta
- rotirajući prioritet, pri kojem svaki kanal u sledećem krugu postaje kanal sa najvišim prioritetom.

BIOS PC računara postavlja fiksan prioritet i nije ga preporučljivo menjati. AT+ računari snabdeveni su još jednim kontrolerom, ovog puta 8237A, koji je vezan kaskadno, tako da je broj novih raspoloživih linija svega tri i one su posvećene šesnaestobitnom prenosu.

IBM-ovi inženjeri, projektujući prvi PC računar, odlučili su da, zarad velikih brzina prenosa, na matičnu ploču ubace **DMA** kontroler. Pri tom su, želeći da uštede na hardveru, dodelili **DMA** kontroleru i ulogu osvežavanja dinamičke memorije. Na liniju **DACK0** vezan je izlaz [illegible] tajmera 8253 i linija se aktivira svakih 15 mikrosekundi, terajući na taj način **DMA** kontroler da osvežava memoriju. Osvežavanje memorije sastoji se u čitanju 256 uzastopnih bajtova negde u memoriji.
Dakle, četiri kanala preko kojih se može vršiti osmobitni prenos:

- Kanal 0, koristi se za osvežavanje memorije.
- Kanal 1, obično je slobodan, mada ga neke mrežne kartice mogu koristiti.
- Kanal 2, koristi ga flopi disk kontroler.
- Kanal 3, može biti korišćen od strane hard disk kontrolera na nekim PC/XT modelima, na sistemima koji nemaju hard disk i na AT i jačim računarima ovaj kanal je slobodan.

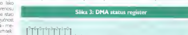
Ako u sistemu imate zvučnu karticu, on da će ona koristiti jedan od slobodnih kanala. Što se tiče drugog **DMA** kontrolera, koji omogućava šesnaestobitni prenos, situacija sa kanalima je sledeća.
Kanal 4, iskorišćen je za kaskadno vezivanje sa prvim kontrolerom, tako da nije slobodan.
Kanali 5-7, slobodni.

Programiranje samog **DMA** kontrolera vrši se preko dostupnih registara, koji su mapirani u I/O memorijskom prostoru. Relevantni registri prvog kontrolera nalaze se na adresama 0000H - 000FH, 0081H-0083H i 0087H, a drugi kontroler zauzima adrese od 00C0H - 00DEH, 0089H-008FH. Precizan spisak svih dostupnih registara dajemo na slici 1.

Da bi pripremili kontroler da prima/šalje podatke od periferije ka njoj, moramo isprogramirati kanal na koji je priključena sama periferija. Najpre maskiramo odgovarajući kanal preko *Single Mask Bit* registra. Zatim upisujemo memorijsku adresu izvora/odredišta podataka u adresni registar, a dužinu podataka (u bajtovima ukoliko je reč o kanalima 1-3 ili u rečima (*words*) ukoliko su u pitanju kanali 5-7) upisujemo u brojač.

Na ovom mestu moramo se malo vratiti u istoriju. Svima je poznata ona nesrećno usvojena granica od jednog megabajta, zbog koje svi danas ispaštamo. Shodno tome, **DMA** kontroleri, koji su tada počeli da se ugrađuju na matične ploče, nisu znali ni za šta iznad 1 MB (20-bitni adresni prostor). Pojavom AT-a, sa novim procesorom koji je imao 4 adresne linije više, dodat je i novi kontroler 8237A, koji se snalazio u novom

Slika 3: DMA status register



...m prostoru. Adresni registri i brojači su šesnaestobitni, pa se postavlja pitanje kako ugurati 20 ili 24-bitnu adresnu u jedan ovakav registar. Odgovor je - nikako, jer DMA kontroler organizuje adresni prostor u stranice.

Naime, prvih 16 bita neke adrese upisuje se u adresni registar odgovarajućeg kanala, dok se ostatak od 4 ili 8 bita upisuje u odgovarajući *Page* registar. U adresni registar i brojač upisuje se prvo niži, pa viši bajt. Rutina koja vrši pretvaranje adrese iz segment:ofset formata u linearni oblik data je na listingu 1. Pre poziva rutine u **DX:AX** nalazi se adresa u obliku segment:ofset, na izlazu se u **DH** nalazi gornjih 4 bita (0-F), a u **AX** se nalazi nižih 16 bita.

Pažljiviji čitaoci će u gornjem pasusu primetiti veliki broj ograničenja, koje pred nas postavlja sam **DMA** kontroler, zbog čega se on gotovo i ne koristi. Kao prvo, pošto brojač ima 16 bita, maksimalna količina podataka koja se može odjednom preneti je 64 Kb, ili 128 Kb ako se radi o 16-bitnom prenosu. To je maksimum, koji u praksi može biti i manji, tj. zavisi od izabrane adrese. Naime, u toku prenosa **DMA** kontroler ne može preći na sledeću stranicu, tako da ako prenos teče od adrese koja se nalazi pri kraju stranice, sigurno nećemo moći da dostignemo veličinu bloka od 64 Kb. Osim toga, osmobitni prenos možemo vršiti samo iz područja prvog megabajta, baš zbog toga što **DMA** kontroler "živi" u 20-bitnom adresnom prostoru.

Pod haubom

Kada smo konačno upisali adrese i dužine u registre, pristupamo određivanju funkcije koju će da vrši kanal. Dajemo pregled funkcija pojednih registara prvog kontrolera, priča je identična i za drugi kontroler.

- **0008H** (*DMA command & status register*). Na ovoj adresi nalaze se komandni i statusni registar. Ukoliko vršimo pisanje na ovaj port pristupamo komandnom registru, u slučaju čitanja dobijamo informacije iz statusnog registra. Preko ovog registra vrši se programiranje samog **DMA** kontrolera (ne kanala!) i nije preporučljivo raditi bilo šta, ako ne znate šta stvarno želite da postignete. Ovako se može isključiti kontroler, uključiti **DMA** transfer na relaciji memorija-memorija, može se promeniti prioritet kanala (objašnjeno ranije) i slično. Na slici 2 dat je kompletan opis. Očitavanjem porta dobijamo statusne informacije o kanalima (da li je neki kanal završio sa prenosom ili je prenos trenutno u toku). Slika 3 to ilustruje.
- **0009H** (*Request register*). Preko ovog registra možemo naterati **DMA** kontroler da počne da obrađuje naš zahtev, tj. da počne sa prenosom, ili možemo prekinuti započeti prenos.
- **000AH** (*Single mask bit register*). Na ovaj način maskiramo određeni kanal, kako bismo ga reprogramirali.
- **000BH** (*Mode register*). Biramo u kojem modu radi kanal. Na ovaj način određujemo da li ćemo vršiti čitanje/pisanje, da li će se početna memorijska adresa uvećavati ili smanjivati tokom prenosa i da li će kontroler vraćati magistrale procesoru posle svakog prenetog bajta ili ne (*single/block mod*).
- **000CH** (*Clear byte pointer flip-flop*). Bilo koji upis na ovaj port rezultuje resetovanjem internog flip-flopa. Tada se svako sledeće upisivanje u neki šesnaestobitni registar (adresni na primer) dekoduje kao upis nižeg bajta. Sledeće upisivanje tretira se kao viši bajt.
- **000DH** (*Master clear*). Upis bilo koje vrednosti na ovaj port resetuje kontroler.
- **000EH** (*Clear mask register*). Upis bilo koje vrednosti demaskira sva četiri kanala.
- **000FH** (*Write mask register*). Preko njega maskiramo/demaskiramo jedan ili više kanala.

Na kraju sledi kratak recept pripreme kanala za rad: najpre zabranimo sve interapte dok programiramo **DMA** i resetujemo interne flip-flopove. Zatim maskiramo odgovarajući kanal popunjavanjem *Single mask bit* registra. Prebacimo početnu adresu iz segment:ofset oblika u linearni format i popunimo adresni i *page* registar. Upišemo i dužinu podataka u brojač. Na kraju, punjenjem *Mode* registra određujemo tip operacije, skidamo masku sa kanala i omogućavamo interapte. Time je kanal spreman za rad.

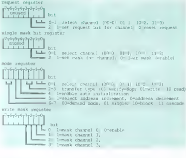
Kao ilustracija gore navedenog, napisana je rudimentarna rutina za programiranje kanala (listing 2 koji možete naći na Sezamu u direktorijumu R:\SOFT). Pomoću nje može se programirati samo prvi **DMA** kontroler. Dodavanje koda koji bi podržao i drugi kontroler, kao i koda koji vrši razne provere parametara (da li se kanal nalazi u opsegu 0-7, da li će u toku prenosa doći do udaranja o kraj stranice i sl.) ne predstavlja veliki problem, a ovde to nije učinjeno zbog nedostatka prostora. Rutine koje su snabdevene sa ovakvim proverama već postoje i nalaze se u javnom vlasništvu. Autor je Christopher M. Box i mogu se naći u više arhiva koje se bave programiranjem SoundBlastera. Jedna od njih je **SBF3.ZIP** (*SoundBlaster Freedom Project*, fajl je **DMA_CODE.ASM**). ■

Autora možete kontaktirati na e-mail adrese pedjak@sezam.co.yu i pedjak@galeb.etf.bg.ac.yu

Listing 1

```
PROC    MakePage
         PUSH    BX
         MOV     BL,DH
         SHR     BL,?
         ADD     AX,?X
         ADC     ?L
         MOV     CH,BL
         POP     BX
ENDP
```

Slika 4



UNIX za neplivače

Iako UNIX više ne spada u egzotičnu temu, ipak postoji veliki broj onih koji ga još uvek ne koriste. Njima tekstovi o tehničkoj strani sistema nisu od velike koristi za formiranje celovite slike o sistemu. Uplovljavanje u UNIX vode može biti neprijatno iskustvo ukoliko se ne zna tačno šta ovaj operativni sistem pruža, za šta ga primeniti, a za šta ne.

Ivan Glišin

Razjasnimo na početku jednu stvar: *UNIX* je operativni sistem koji nije nastao na IBM PC kompatibilnim računarima, niti je pravljen kao zakrpa za *MS DOS*. Pretpostavljam da *UNIX* znalce ova konstatacija može slatko nasmejati, ali iz pitanja ljudi koji su samo čuli za *UNIX* i nemaju nikakva iskustva sa njim, zaključujem da nisu svi baš sasvim na čisto o čemu je reč. Zabune su razumljive s obzirom na mogućnost instaliranja na IBM-PC kompatibilne računare, a potpunu zbrku prave *MS-DOS* emulatori kao i mogućnost instaliranja nekih verzija sistema na postojeću *MS-DOS* particiju. Zapravo, *UNIX* je samostalan i potpuno drugačiji operativni sistem koji ne radi samo na PC platformama. Prve verzije nastale su pre nego što se pojavio prvi mikroprocesor, a daleko pre vreme na najprimitivnijih kućnih i personalnih računara.

Kada kažemo *UNIX*, mislimo na celu familiju sličnih operativnih sistema koji su izvedeni iz zajedničkog pretka. Doduše, *UNIX* je registrovano ime kompanije *Unix System Laboratories (USL)* koja je vlasništvo kompanije *AT&T*, odakle je sistem i potekao, ali ćemo ga mi koristiti kao opšte ime za sve varijante sistema iz porodice.

Istorijski razvoj

UNIX je začet 1969. godine u *Bell* laboratorijama na računaru *DEC (Digital Equipment Corporation) PDP 7* na kome su *Ken Thompson* i *Dennis Ritchie* sa saradnicima započeli sa radom. Sledećih desetak godina se razvoj odvijao na *DEC PDP-11* a zatim i *DEC VAX* sistemima. Interesantno je pomenuti da čak i danas u reklamnom materijalu kompanije *Digital*, uz mašine koje rade pod *DEC ULTRIX* (*Digital*-ova verzija *UNIX* sistema) stoji da je baš na njihovim računarima nekada davno *UNIX* sistem i nastao.

Bitna promena u razvojnoj koncepciji se dogodila 1973. godine kada je sistem napisan u jeziku C. Jezik C je nastao iz jezika BCPL, jezika B, a razradili su ga *Brian Kernighan* i *Dennis Ritchie* specijalno za potrebe razvoja ovog operativnog sistema. Od tada je C razvijan zajedno sa *UNIX* sistemom i oni su neraskidivo povezani. Iako je specifične namene i sa teorijskog aspekta predstavlja prilično neugodan jezik, C doživljava niz proširenja i svoju standardnu varijantu 1979. godine. Vremenom postaje izuzetno popularan među sistemskim programerima, pa je C kompajler pisan i za druge platforme, čime je automatski obezbeđeno i migriranje *UNIX* sistema.

Od 1975. godine *UNIX* izlazi iz *Bell* laboratorija i počinje da se širi prvenstveno po univerzitetima. Primena u poslovnom svetu je kasnila, jer je tržište bilo dobro podržano od strane giganata za poslovnu informatiku, među kojima su bili, recimo, *Control Data, Burroughs, Digital* ili *IBM*. Sa svojim mašinama, operativnim sistemima, aplikacijama i izvanrednom podrškom nudili su poslovnom svetu sve što je bilo potrebno. Ti sistemi su bili izuzetno stabilni i sigurni, što je neophodno za primenu u sferi poslovne obrade podataka, ali prilično neupotrebljivi za primenu u nauci i inženjeringu. Upravo je *UNIX* odigrao ulogu kao bitan sistem za koji se počinju razvijati inženjerske i naučne aplikacije, posebno kada je u pitanju rad na grupnim projektima i upotreba računarske grafike. Tek sredinom osamdesetih on postaje viđeniji u svetu poslovnih ljudi, mada je i dalje popularniji u univerzitetskom i naučno-inženjerskom svetu.

Od kraja sedamdesetih *UNIX* počinje da se razvija u više pravaca, naročito zahvaljujući portabilnosti delova sistema pisanih C jezikom. Mnoštvo timova pri raznim institucijama razvija mali deo sistemski zavisnog koda (jezgro sistema) za specifične mašine, a zatim koriste *AT&T* svoje ili druge komponente sistema pisane u C jeziku, prevode ih i za kratko vreme dobijaju verzije *UNIX* sistema za vrlo različite hardverske platforme. Naravno, ovaj razvoj je obuhvatao i dodavanje novih i modifikaciju postojećih komponenti sistema, što je bilo lako s obzirom na upotrebu C jezika, čime su se stvorile male razlike između različitih razvojnih pravaca. Ovo bi naizgled moglo voditi ka potpunom razilaženju, ali se zapravo desilo da su svi jedni od drugih preuzimali zgodna unapređenja i dodavali ih kod sebe. Tako su svi *UNIX* sistemi, iako različiti, ipak toliko slični među sobom da prelazak sa jednog na drugi sistem ne predstavlja bitnu promenu. Aplikacije se mogu preneti sa jedne platforme na drugu uz minimalan trud, tako da danas na svom sistemu možete imati sve ono što postoji i na drugim sistemima, iako su oni hardverski potpuno različiti. Ovo je jedna od ključnih stvari koja je doprinela njegovoj velikoj popularnosti.

Pravci razvoja

Za poslednjih dvadesetak godina nastalo je na hiljade varijanti sistema. Međutim, može se govoriti o dva bitna pravca razvoja, kojih svi nastoje manje ili više da se pridržavaju. To su *AT&T* i *BSD*.

Prvi pravac je izrastao iz *UNIX System V* (pet) kompanije *AT&T*. Sistem V je razvijen iz *UNIX System III* koji je bio prva komercijalna verzija *UNIX* sistema *AT&T*-a izbačena na tržište 1982. godine. *System V* prezentiran je 1983. godine i doživeo je niz unapređenja koja se zovu izdanja (*release*). Radi kompaktnosti zapisa cela serija *System V* izdanja se u literaturi obično navodi u skraćenom obliku, tako da se recimo *UNIX System V Release 2* obično piše kao *SVR2*. U nastavku ćemo se pridržavati ove notacije. Dosta rasprostranjena verzija je *SVR3.2*, ali je interesantna i *SVR4* iz 1988. godine koja ujedinjuje elemente *SVR3.2*, *4.3BSD* i *SunOS* na liniju razvoja cele *System V* familije. Neke od poznatijih *System V* varijanti koje se mogu sresti na našim prostorima su: *IBM AIX* (SVR3.2), pre svega za *IBM RS/6000 RISC* sisteme, *Apple A/UX* (SVR2) za *Macintosh* sisteme sa *Motorola* 680x0 procesorom, *Hewlett Packard HP UX* (SVR3.2) za *HP9000* sa MC680x0, *Silicon Graphics International IRIX* (SVR3.2 sa dosta BSD proširenja) za *MIPS R4000* familiju procesora, *Linux* (baziran na *System V*, ali zbog dosta brzog i fleksibilnog razvoja pokriva specifičnosti različitih varijanti sistema i različite platforme), *The Santa Cruz Operation SCO XENIX* (SVR2) i *SCO UNIX* (SVR3.2) za 80386 sisteme. U zagradi su navedena izdanja na kojima su konkretne verzije bazirane, što znači da su sa njima uglavnom saglasni, ali su svi proizvođači modifikovali sistem u skladu sa njihovim specifičnim potrebama. Tako navedeni podatak treba shvatiti kao orijentaciju, doduše dosta preciznu, ali treba očekivati manja ili veća odstupanja među adaptacijama istih izdanja od strane različitih proizvođača.

Za drugi pravac je zaslužan *Berkeley Software Distribution* (BSD). Razvijan je u vrlo dinamičnom akademskom i inženjerskom okruženju, pa su se prirodno pojavila mnoga unapređenja koja su uglavnom prenošena i na *System V*. Prva bitna varijanta je bila 2.0BSD iz 1978. godine na *DEC PDP 11*, ali od pravog značaja su 4.xBSD s početka osamdesetih. Interesantno je da su verzije 4.2BSD i 4.3BSD donele podršku za *TCP/IP* na *Ethernet* (IEEE 802.3) fizičkom sloju i *NFS* (*Network File System*), što je definitivno otvorilo prostor *UNIX* sistemima za upotrebu u mrežnim i distribuiranim okruženjima. Kao i *System V*, BSD takođe postoji u nizu varijanti. Kod nas se uglavnom mogu sresti *Berkeley Software Distribution NetBSD* (4.3BSD) kao PD varijanta za 80386 sisteme, *Sun Microsystems SunOS* i *Solaris* (4.3BSD) za MC680x0, SPARC i 80386, *Digital Equipment Corporation ULTRIX* (4.2BSD, 4.3BSD) za *DECstation* i *DEC VAX* sisteme.

Šta nam UNIX donosi?

UNIX je *multi tasking* operativni sistem, što znači da u isto vreme može izvršavati više poslova koji se na *UNIX* sistemu zovu procesi. Na primer, možete startovati nekoliko procesa u pozadini i, ne čekajući da se oni završe, raditi dalje interaktivno. Pod ovim se podrazumeva da i neki program sam može aktivirati više svojih procesa koji će se izvršavati konkurentno (*multi threading*).

U pitanju je višekorisnički (*multi-user*) operativni sistem, što znači da više od jednog korisnika može koristiti sistem u isto vreme. To je normalna posledica prethodne osobine. Tako se resursi efikasnije koriste, korisnici mogu raditi na zajedničkim problemima i deliti programe i podatke. Takođe, održavanje takvog sistema je efikasnije jer je centralizovano.

UNIX ima ugrađene mehanizme zaštite programa i podataka. Ukoliko se pažljivo konfiguriše, gotovo je nemoguće uništiti ili ugroziti zaštićene delove sistema čak i uz priličan trud, a o slučajnosti da ne govorimo. Novije varijante *UNIX* sistema obezbeđuju i aktivnu zaštitu koja, pored postojeće pasivne zaštite fajlova na disku, obezbeđuje mehanizam praćenja rada korisnika i onemogućava ga da uopšte izvede bilo koju akciju za koju nije ovlašćen.

Od velikog je značaja i činjenica da je *UNIX* izuzetno pouzdan sistem. To ne znači da računar koji radi pod njim nikada neće pasti, ali je to izuzetno retko. Čak i kada se desi da sistem iz bilo kog razloga krahira, *UNIX* obezbeđuje mehanizme za oporavak, pa je izuzetno retko da se uz pravilno instaliran, podešen i održavan sistem napravi značajna šteta. Takođe, svaki proces ima svoju memorijsku zonu i sistem mu ne dozvoljava da koristi bilo šta drugo. Tako tipične greške u programiranju koje na *MS DOS* sistemima izazovu blokiranje računara i potrebu da se računar ponovo startuje, *UNIX* sistem detektuje, prekine proces, sredi stanje i obavesti korisnika šta je po sredi. Ukoliko u isto vreme na sistemu radi još nekoliko korisnika, niko od njih neće ni primetiti da se bilo šta dogodilo.

Moramo pomenuti i izuzetno efikasnu i fleksibilnu organizaciju podataka na disku (*file system*) i programe za komunikaciju sa periferijskim jedinicama (*device drivers*). Recimo, podaci na disku se uvek vide kao jedinstvena hijerarhijska struktura bez obzira da li je cela na jednom disku, da li je na dva ili više lokalnih diskova, da li je delom na lokalnom disku a delom na disku susedne mašine ili mašine udaljene stotinu kilometara. Takođe periferijskim uređajima se obraćamo isto kao i bilo kom fajlu, pa u njih pišemo i iz njih čitamo podatke na isti način na koji to radimo sa bilo kojim drugim fajlom.

UNIX je veoma portabilan sistem. Veći deo sistema je napisan u programskom jeziku C koji je i nastao kao alat za razvoj *UNIX* operativnog sistema. Osim jezgra sistema (*kernel*) svi ostali delovi su pisani u C jeziku. Tako, da bi se napravio *UNIX* sistem za neki računar, potrebno je napisati jezgro u mašinskom jeziku sistema, ili kombinaciji C jezika i mašinskog jezika sistema zbog specifičnih zahteva konkretnog hardvera, a zatim prevesti iz C jezika programe kao što su komande sistema, editori teksta, komunikacioni paketi, sistem za elektronsku poštu, sistem za formatiranje teksta i slično. Dakle, o konkretnoj mašini se brine samo kada se piše jezgro sistema, ostatak je uglavnom potpuno nezavisan od konkretne opreme i zahteva minimalne ili nikakve izmene. Tako se prebacivanje sistema na potpuno novu platformu obično može završiti za nekoliko meseci.

Zahvaljujući pomenutoj portabilnosti *UNIX* je veoma rasprostranjen sistem. Postoje hiljade verzija sistema i milioni instalacija širom planete. Nema ni jednog ozbiljnog proizvođača računarske opreme koji ne podržava neku varijantu *UNIX* sistema, bilo sopstvenu bilo tuđu, kao na primer *Digital* (*ULTRIX*), *Hewlett-Packard* (*HP UX*), *IBM* (*AIX*), *Honeywell* (*LPI UNIX*), *Apple* (*A/UX*), *CRAY* (*UNICOS*) i tako redom. Pored ovoga na raspolaganju je i nekoliko *UNIX* sistema nezavisnih proizvođača namenjenih IBM PC kompatibilnim računarima, pre svega *SCO XENIX/UNIX*, kao i nekoliko besplatnih, slobodno dostupnih verzija kao što su *FreeBSD* ili *Linux* koji kod nas postaju prilično popularni.

UNIX ima izuzetan skup komandi i uslužnih programa. Posebno je interesantna mogućnost vezivanja izlaza iz jedne komande na ulaz u drugu (*pipeline*) što daje izvanredne mogućnosti kombinovanja komandi. Neke mogućnosti kojima obiluju novije verzije *MS-DOS* sistema, kao i zamena dopuna za njega (4DOS, DRDOS, NU) kojima se njihovi korisnici oduševljavaju, na *UNIX* sistemima postoje često više od deset-petnaest godina i sasvim su uobičajena stvar. Pored toga, sistem dolazi sa veoma jakim alatima za programiranje. Standardno razvojni sistem se, pored kompajlera za jezik C, sastoji i od niza korisnih programa za kontrolu ispravnosti koda, generisanje liste referenci, liste korišćenih simbola, leksičku analizu, parsiranje, ulepšavanje C koda, kontrolu verzija programa, dibagera i još mnogo vrlo korisnih alata.

Sistem se lako proširuje novim programima, pre svega uslužne prirode. Zahvaljujući razvojnom sistemu koji je standardni deo svakog *UNIX*-a (ako već nije instaliran, svakako ga proizvođač ima u ponudi) bilo koji proizvođač softvera može uz minimalne modifikacije prilagoditi izvorni kod nekog programa bilo kom sistemu. Takođe, *UNIX* ima milionsku populaciju programera koji svakodnevno pišu programe kojima rešavaju svoje probleme i za koje smatraju da bi bili korisni i drugima. Zato izvorne kodove distribuiraju putem računarskih mreža i tako ih stavljaju na raspolaganje svim korisnicima *UNIX* sistema.

I na kraju, *UNIX* je standardno snabdeven kompletnom dokumentacijom. To znači da u bilo kom trenutku možete sa terminala zatražiti uputstvo za bilo koju komponentu sistema i dobićete isto ono uputstvo koje se nalazi i u štampanoj dokumentaciji. Zapravo, na *UNIX* sistemu za formatiranje teksta je i pripremljena kompletna dokumentacija za štampu, a tako pripremljen materijal se isporučuje na disku i na raspolaganju je svim korisnicima preko terminala.

Kada koristiti UNIX?

Za jednog korisnika koji drži svoju mašinu na sopstvenom radnom stolu i sam je koristi sasvim je dovoljan *MS DOS* u kombinaciji sa *MS-Windows* okruženjem, posebno ukoliko nema potrebu da intenzivno razmenjuje podatke sa drugim korisnicima. Međutim *UNIX* se svakako nameće kao sistem koji je interesantan u sledećih nekoliko slučajeva:

- Ukoliko podatke treba da koristi više od jednog korisnika, *MS-DOS* postaje problematičan. Jasno je da se *UNIX*, *VAX/VMS* ili *IBM/VM* nameću kao moguća rešenja, ali se *UNIX* zbog rasprostranjenosti hardverskih platformi na kojima radi i relativne ne

zavisnosti od jednog proizvođača po pitanju održavanja nameće kao realno rešenje. Lokalne mreže PC računara mogu donekle rešiti problem, ali je u tom slučaju pitanje održavanja celog sistema dominantno, pa je u nekim slučajevima efikasnije imati jedan ili nekoliko višekorisničkih računara i mrežu terminala. Za većinu poslovnih primena ovo je odlično ili čak jedino prihvatljivo rešenje.

• Ukoliko pravite BBS, *file server* ili sličan sistem kod kuće ili u preduzeću, *UNIX* je vrlo interesantan jer je u startu višekorisnički, ima podršku komunikaciji preko modemskih linija a lako se vezuje i razmenjuje podatke i sa drugim *UNIX* sistemima.

• U slučaju da vaš računar treba da održava deo mreže, *UNIX* je izvanredan izbor jer je snabdeven podrškom za najraznovrsnije komunikacione protokole, tako da je uz relativno jeftinog hardvera moguće graditi i održavati izuzetno složene komunikacione strukture.

• Ako se bavite razvojem softvera, *UNIX* je odličan sistem jer je snabdeven svim alatima za programiranje kojima je i on sam napravljen. Tokom svog razvoja je dobio univerzalnost i postao primenljiv u svim oblastima, ali je i danas standardno opremljen izuzetno snažnim alatima za programiranje i pruža velike mogućnosti i udobnost prilikom razvoja softvera.

• Ukoliko se bavite primenom u inženjeringu ili nauci, *UNIX* je u kombinaciji sa *X Window* okruženjem, kao standardnim grafičkim okruženjem za *UNIX* sisteme, izvanredno efikasan. Na primer, aplikacije iz oblasti računarskog projektovanja, ekspert sistema, simulacija i vizuelizacije podataka su za *UNIX/X Window* platforme razvijene do granica neverovatnog. Svaki pokušaj poređenja pomenutih kategorija *X Window* aplikacija sa odgovarajućim rešenjem za bilo koju varijantu *MS-Windows* okruženja je najblaže rečeno neukusan.

Šta izabrati?

Do pre samo nekoliko godina, krajem osamdesetih, operativni sistem *UNIX* je bio relativno redak na našim prostorima. Iako se mogao sresti na fakultetima ili u nekim preduzećima kao višekorisnički operativni sistem, njegova upotreba u kućnim uslovima, dakle na malim PC kompatibilnim računarima, bila je gotovo zanemarljiva.

Premda su postojale verzije sistema za procesor 80286, različite varijante *UNIX* sistema su postale dostupne i za kućnu upotrebu tek pojavom procesora 80386 i pada cene kompletnog hardvera računara građenih oko 80386 i jačih, a pre svega povećanjem kapaciteta diskova. Kod nas je intenziviranje upotrebe na kućnim računarima uzelo maha tek 1993/94-te, posebno sa pojavom nekomercijalne verzije sistema pod nazivom *Linux*, koja se kod nas jako brzo širi, a i intenzivnijom upotrebom *SCO XENIX/UNIX* sistema na univerzitetu i u preduzećima. Danas je moguće odvojiti deo diska za *UNIX* sistem, dok je na drugom delu diska moguće zadržati *MS-DOS* sistem, tako da na istoj mašini imamo dva danas prilično značajna i vrlo prisutna operativna sistema (recimo da *MS DOS* možemo častiti tim imenom). Ukoliko želite da okusite malo (ili malo više) čari *UNIX* sistema, *SCO UNIX* ili *Linux* se nameću kao rešenje. Svaki od njih ima svoje za i protiv.

SCO UNIX je komercijalna verzija kompanije *The Santa Cruz Operation* i izuzetno je dobro podržana od strane proizvođača. *SCO* ide u pravcu primene *UNIX* sistema u poslovnom okruženju, pa je logično odlučiti se za njega ukoliko imate ideju da razvijate baze podataka i softver za primenu u industriji i finansijama. Radi se o ozbiljnom, stabilnom i sigurnom proizvodu, premda pomalo nezgrapnom po pitanju sistemskih zahteva i održavanja. Za *SCO UNIX* ćete naći niz kvalitetnih paketa kao što je recimo *ORACLE*, sistem za upravljanje bazom podataka koji predstavlja svetski standard. Nažalost, od podrške proizvođača mi nemamo nikakve koristi pa je taj razlog za izbor, iako presudan u inostranstvu, kod nas beznačajan. Međutim, postoji podrška od strane mnoštva naših ljudi koji rade sa proizvodima *SCO*, razvijaju softver za njih ili ih samo održavaju, tako da se svakako nećete osećati kao na pustom ostrvu ukoliko se odlučite za njega. Međutim, on nije sjajno rešenje za programerske, inženjerske, naučne ili obrazovne potrebe. Nedostatak ili teška dostupnost aplikacija ovog tipa i niza programskih jezika delimično zatvaraju vrata tom segmentu primena.

Linux je nekomercijalna verzija *UNIX* sistema koja se praktično razvija na mreži, pa su tako u njegov razvoj uključeni ljudi sa cele planete. Koliko mi je poznato, nijedan softverski proizvod takve kompleksnosti nije obuhvatio toliku prostornu razuđenost niti je bio internacionalizovan u tolikoj meri. Ovo garantuje izuzetnu podršku i brz razvoj, pa tako *Linux* pretenduje da ujedini dobre osobine svih bitnih *UNIX* varijanti koje su se do sada pojavile. Zbog svoje fleksibilnosti i snabdevenosti velikim brojem interesantnih programa, *Linux* je postao izuzetno popularan kao kućni *UNIX* i kao *UNIX* za univerzitetsko, inženjersko i naučno okruženje. *Linux* je opremljen izvanrednim alatima, nizom programskih jezika, podrškom za najraznovrsnije periferije, izuzetno je dobar kao deo mreže, ima sjajnu podršku za *X Window* grafičko okruženje, što mu sve skupa daje izuzetnu primenljivost za najrazličitije svrhe. Međutim, nedostatak podrške od strane velikih softverskih kuća, nepostojanje bilo kakvih klauzula koje garantuju zaštitu investicije, a posebno nemogućnost kupovine sa registracijom (što kod, recimo, *SCO* povlači podršku u obliku instalacije, obuke osoblja i stalnog održavanja) odbija korisnike iz poslovnog sveta.

Za koji god sistem da se odlučite, imajte na umu da se radi o složenom operativnom sistemu koji treba održavati, čije startovanje može da potraje, na koji se morate prijaviti kao korisnik ako želite bilo šta da radite i koji se na kraju ne može isključiti na dugme, već se mora startovati procedura za obaranje sistema, koja takođe može potrajati. Ovo napominjem zbog toga što su mnogi koji su stavili *UNIX* brzo odustali od njega jer su očekivali da će automatski dobiti nešto mnogo bolje od sistema *MS DOS* i da pri tome neće morati da menjaju stečene navike. To je pogrešna pretpostavka, pa zato ukoliko vaš PC palite i gasite deset puta dnevno da biste pozvali neki BBS modemom, odigrali partiju tetrisa ili otkucali par redova teksta, ostavite *MS DOS* na mašini. Nemojte prelaziti na *UNIX* ukoliko nemate prave potrebe za njim i ne znate čime da ga zaposlite.

Dalji razvoj

Šta će se dešavati sa *UNIX* om u narednih desetak godina nije teško naslutiti, makar u grubim crtama. Pre svega, razvoj se već odvija u pravcu usklađivanja njegovih verzija među sobom, delom kao rezultat tržišnih kretanja a delom kao rezultat dogovora i postavljanja standarda. Od svog nastanka sistem je neprekidno unapređivan, postajući tako siguran i sposoban za ozbiljne primene. Tako na primer na *UNIX* se sada dodaju razni stepeni aktivne zaštite sistema u celini, što je neophodno da bi uspešno uplivao u vode poslovne informatike i postao siguran mašina na mreži. Otvorenost sistema vodi ka brzom razvoju i krpljenju rupa koje se pojavljuju tokom eksploatacije, pa je tako garantovano da *UNIX* neće ostati bez podrške, jer nije striktno vezan za jednog ili grupu proizvođača. Na kraju, nove tehnike realizacije koje podržavaju mikrokernel arhitekturu i primenu algoritama za balansiranje opterećenja uz korišćenje simetričnog multiprocesinga otvaraju izvanredne mogućnosti efikasnog iskorišćenja hardverskih resursa koji su nam na raspolaganju i novih hardverskih koncepcija čije vreme tek dolazi. Zbog svega toga, *UNIX* je svakako sistem koji je zaslužio pažnju i koji ćemo narednih godina sve češće sretati. ■

NEUREL '95

Posle višegodišnje pauze na Elektrotehničkom fakultetu je 29. i 30. septembra održan treći seminar pod imenom "Primena neuralnih mreža u elektrotehnici". U dva nastavka donosimo detaljan izveštaj.

Ivan Sajić

Tokom dva cela dana predstavljeno je preko 30 radova u kojima je učestvovalo gotovo 50 jugoslovenskih stručnjaka. Organizacija se pokazala veoma uspešnom, počev od stručnog nivoa celog seminara i pozvanih učesnika, preko kvalitetnog zbornika radova, pa od prozaičnih stvari kao što su pauze na svaka dva sata sa neophodnim i ukusnim posluženjem za iscrpljene učesnike.

Sa zadovoljstvom se može reći da je ova oblast u našoj sredini ne samo izašla iz faze dečijih bolesti od kojih je patila pre 7-8 godina, nego i da je u uslovima blokade znatan broj naših stručnjaka uspeo i da prati svetska istraživanja u oblasti informacija, ali i da predstavi zanimljive i originalne rezultate.

Radovi su tematski bili podeljeni na tri grupe - sednice. Prva grupa radova, pod zajedničkim imenom "Opšti principi rada NM", odnosila se na opšte aspekte NM - arhitekturu i algoritme obučavanja. Drugu grupu, pod imenom "Primene NM" činili su radovi koji su se bavili robotikom, upravljanjem i telekomunikacijama. Treću grupu, "Realizacije i biološki aspekti" činili su radovi vezani za biomedicinske aspekte i implementaciju NM. Pokušaćemo da pobliže predstavimo ova istraživanja.

Opšti principi rada NM

Prvi rad (Milan Milosavljević, Srđan Stanković) predstavljen na ovoj sednici dao je opšti prikaz pojmova. Dat je opis neurona, veza između njih i aktivacione funkcije. Prikazana je klasifikacija NM u odnosu na tok signala, reprezentaciju znanja i u odnosu na način obučavanja. Zatim su navedene neke osobine višeslojnih perceptrona, a na kraju je predstavljeno nekoliko značajnijih tipova NM i njihovih primena: Hopfildova (diskretni optimizacioni problemi), Hamingova, Kohonenova i Karpenter-Grozbergova (procesiranje signala).

Tema sledećeg rada bila je **Obučavanje višeslojnih perceptrona** (Srđan Stanković, Milan Milosavljević) metodom povratnog prostiranja greške (*BackPropagation*). Data je detaljna analiza konvergencije obučavanja ovih NM, pa je pokazano da je od značaja i izbor aktivacione funkcije. Istaknuto je da se algoritmi obučavanja zasnivaju na postupcima stohastičke optimizacije. Razmatrani su predlozi za povećanje brzine obučavanja (konvergencije) uvođenjem matričnog računa, mada je konstatovano da je uvođenje matrica povezano sa povećanjem vremena računanja. Razmotreni su i višekoračni algoritmi obučavanja po kojima promena sinaptičkih težina ne zavisi samo od njihove poslednje vrednosti, već i od njihovih vrednosti u prethodnim trenucima. Pokazano je i da je poznati algoritam BP sa "momentom" specijalan slučaj višekoračnih algoritama.

Bisku temu imao je i rad **Algoritmi za brzo učenje višeslojnih perceptrona baziranih na proširenom Kalmanovom filtru** (Duško Katić, Srđan Stanković). Pomenuto je da je najpopularniji algoritam učenja - BP zasnovan na gradijentnom metodu ili metodu najbržeg spusta. Naime, ako se greška koju pravi NM shvati kao funkcija u prostoru čije su koordinate sinaptičke težine, onda se BP obučavanje svodi na minimizaciju ove funkcije kretanjem naniže u pravcu najvećeg nagiba. Pokazuje se da algoritmi koji se baziraju samo na informaciji o prvom izvodu (nagibu) ne omogućuju dovoljno brzo smanjenje greške (obučavanje). Brzina konvergencije i tačnost predloženog algoritma oprobani su na klasičnom XOR problemu (obučiti NM sa dva ulaza da generiše XOR funkciju). Predloženi algoritam (EKF) se u svakom smislu pokazao boljim od standardnog BP algoritma.

I rad **Jedan konvergentan algoritam obučavanja troslojne NM** (Jelena Simanović, Srđan Stanković, Milan Milosavljević) bavio se problemima i mogućim poboljšanjem algoritma BP. Pokazano je da BP ne mora da konvergira sa sigurnošću, a da gradijent može da dostigne veliku vrednost. To ima za posledicu oscilovanje težinskih koeficijenata, pa i divergenciju (neograničeni skok) greške. Pomenuto je da BP sa momentom uglavnom pokazuje bolje ponašanje, ali ne uvek. Predložena je i modifikacija BP algoritma čija se svojstva poboljšavaju kada se gradijent podeli kvadratom norme vektora težina uvećane za jedan. Efikasnost tri algoritma - originalnog BP, BP sa momentom i BP sa umanjenim gradijentom - razmatrana je na tri primera: razlikovanju dve preklapajuće Gausove raspodele, XOR problemu i aproksimaciji polinomalne funkcije. Predloženi algoritam (sa umanjenim gradijentom) se u svim slučajevima pokazao kao bolji.

Poslednji rad posvećen obučavanju bio je **Algoritam obučavanja NM usmeren ka minimizaciji Bajesove greške odlučivanja** (Momčilo Teovanović, Milan Milosavljević). Posmatran je problem trgovačkog putnika (treba obići sve unapred zadate gradove, a da ukupni put bude što kraći). Ovaj problem može se rešavati Nilsonovim A algoritmom, jer on zahteva definisanje klasifikujuće heurističke funkcije, koja samo za gradove na najkraćoj putanji daje vrednost 0. Sada pogodno obučena NM treba da realizuje takav klasifikator čija će diskriminaciona funkcija biti adekvatna pomenutoj heurističkoj proceni perspektivnosti da dati grad leži na najkraćoj putanji. Korišćen je opet BP algoritam, ali sa modifikovanim proračunom greške (*Modified Mean Squared Error*). Pokazano je da ovako modifikovani BP algoritam dostiže referentne klasifikatore bazirane na statističkom prilazu (*k-Nearest Neighbors*). Sa druge strane, modifikovana greška opada brzo, a ne asimptotski kao kod klasičnog BP algoritma.

Rad **Uticaj ograničenog skupa podataka za obučavanje na performanse NM** (Milan Marković, Milan Milosavljević) posmatrao je opšti problem klasifikacije i primenu statističkih metoda (k-NN, AutoRegresija, Kvadratni Klasifikatori) u odnosu na metode NM. Izvedeni su eksperimenti sa prepoznavanjem kontura na digitalizovanim fotografijama koje su podvrgnute različitim transformacijama (pomeranja, rotiranja). Razmatrane su NM sa jednim i dva skrivena sloja. Zaključak rada je da mreže sa dva skrivena sloja zahtevaju za obučavanje neuporedivo veći broj podataka od mreža sa jednim skrivenim slojem. Za isti broj podataka, mreže sa jednim skrivenim slojem pokazuju bolje rezultate i stabilniji rad. Preporučeno je da broj iteracija učenja bude prilagođen karakteristikama raspoloživih eksperimentalnih podataka.

Hibridi i trajektorije

U prvom od dva rada posvećena hibridnim sistemima veštačke inteligencije, **Rad sa nepouzdanim podacima u neuro-ekspertnim sistemima** (Ljiljana Stoiljković, Nenad Stojanović, Živko Tošić) razmatrana je mogućnost prenošenja logičkih pravila iz jednog ekspertnog sistema u NM. Pokazano je da se sva pravila mogu prikazati u formi tzv. *Horn* ovih klauzula: Ako je A1 i A2 ... onda je B. U tom smislu preuzet je određeni oblik prenosne funkcije. Pokazano je da početno znanje određuje početnu strukturu mreže i težine veza NM. Tako formirana NM može da stiče nova znanja i prečišćava stara znanja empirijski, na osnovu primera. Ovako revidirano znanje, ugrađeno u NM, moguće je dekodirati u skup pravila. Kao

primer uzet je ekspertni sistem za dijagnostikovanje kvara pumpi.

Drugi rad na ovu temu je **Generisanje pravila iz neuronskih mreža** (Nenad Stojanović, Ljiljana Stoiljković, Živko Tošić). Konstatovano je da je najveća slabost pristupa veštačkoj inteligenciji preko NM u teškoj interpretaciji prikupljenog znanja. Način na koji je mreža obavila postavljeni zadatak ostaje tajna zbog nemogućnosti da se prati semantika promena težina i aktiviranja pojedinih neurona. Metode izvlačenja znanja iz NM omogućavaju da mreža sama sebe objasni. Ovako izvučena pravila mogu se ugrađivati u pojedine module ekspertnih sistema i tako dobijati hibridne sisteme veštačke inteligencije. Konačno, pomenuta su četiri metoda za izdvajanje pravila iz NM, a predstavljen je i jedan algoritam.

U predavanju **Učenje trajektorija primenom diskretnih rekurentnih mreža lokalnih procesnih elemenata** (Branimir Todorović, Miomir Stanković) objašnjeno je da se problemi identifikacije i kontrole mogu posmatrati kao problemi učenja ili generisanja trajektorije pri čemu se vrši optimizacija nekog kriterijuma. Na primer, kontroler mora da obezbedi seriju kontrolnih akcija tako da minimizira razliku između trenutnog izlaza i izlaza referentnog modela. U radu je razmatrana primena rekurentnih mreža pri učenju trajektorija. Dokazana je ekvivalentnost ovakvih mreža i rekurentnih *fuzzy* modela, a zatim je, primenom diskretnog varijacionog računa, izveden algoritam učenja trajektorije primenljiv na oba sistema. Pomenimo i to da se *fuzzy* logika bavi formalnim principima nepreciznog (približnog) zaključivanja na osnovu nepreciznih tvrđenja.

Celularne NM

Na kraju ove sednice odvajamo nešto veći prostor za rad **Celularne NM: strukture, realizacije, primene** (Branimir Reljin, Pavle Kostić). CNM su relativno nova klasa veštačkih NM koje su zbog svoje planarne strukture i lokalne povezanosti ćelija veoma pogodne za VLSI realizaciju. CNM su rođene 1987. godine u laboratoriji profesora *Leon*-a *Chu*-a u Berkliju, a predstavljene su široj naučnoj javnosti naredne godine. Nepune dve godine posle originalnog rada, održan je i prvi međunarodni skup o CNM na koji je prijavljeno stotinak autora. Nakon toga su 1992. i 1994. održana još dva međunarodna skupa. Organizacija IEEE objavila je i specijalni broj časopisa posvećen CNM.

CNM mreža sastoji se od velikog broja ćelija koje su lokalno povezane sa svojim susedstvom na identičan način. Veza se ostvaruje sinapsama čije vrednosti određuju ponašanje mreže. Takva struktura, koja je uobičajeno planarna i pravougaone forme (inače može biti još i trougaone ili šestougaone) veoma je pogodna za VLSI realizaciju. Mada ćelije ne moraju biti naročito brze (reda mikrosekunde), zahvaljujući paralelnom radu vreme rada ukupne mreže je veoma kratko (reda mikrosekunde), uglavnom nezavisno od broja ćelija. Danas se proizvode čipovi pogodni za rešavanje velikog spektra raznih problema. Kombinovanjem analogne strukture CNM sa logičkim operacijama ugrađenim u čip, dobija se analogička mašina nazvana CNM univerzalna mašina – analogni ekvivalent digitalnom mikroprocesoru. Ovakva struktura može se isprogramirati za željenu primenu. Prvi CNM čip, realizovan u dvomikronskoj CMOS VLSI tehnologiji i testiran, pokazao je brzinu od blizu hiljadu milijardi analognih operacija u sekundi. Može se očekivati da će se, slično svojevremenoj ekspanziji PC računara, razviti nova industrija bazirana na ovakvim čipovima, koja će proširiti mogućnosti PC računara i dovesti do stvaranja neke vrste personalnih superkompjutera.

U ovom radu je zatim dat pregled struktura CNM, počev od Čuovog rada do kasnijih modifikacija. Zatim su razmotrene njene realizacije, a kraj rada je posvećen najznačajnijim primenama. Struktura CNM mreže sugeriše, zbog svoje planarnosti, pre svega, primenu u obradi slike: za detekciju linija, uklanjanje šuma, izdvajanje oblika ili karakterističnih detalja, što je iskorišćeno za brzo prepoznavanje karaktera. CNM su primenjivane i u procesu ditheringa (predstavljanju nivoa osvetljenja gustinom crno-belih tačkica). CNM nalaze primenu u medicini – u mamografiji (prepoznavanju kancerogenog tkiva dojke), za predikciju kretanja ruke (primena u robotici ili rehabilitaciji invalida). Mogu se uspešno koristiti i u rešavanju sistema linearnih jednačina/nejednačina, raznim optimizacionim problemima, analizi mehaničkih vibracija, kao i rešavanju nelinearnih problema provođenja toplote u pločama različite geometrije. Najzad, zbog izvesne sličnosti sa biološkim nervnim sistemom, CNM mogu veoma dobro da modeluju neuroanatomiju – model mrežnjače, lateralne inhibicije u čovečjem vizualnom sistemu.

Primene NM

Predavanje Duška Katića **Neuronske mreže u robotici** veoma detaljno prikazuje problematiku upravljanja robotima. Objašnjeno je kako konvencionalne metode upravljanja nisu dovoljne za ostvarenje kvalitetnih performansi i današnjih zahteva. Od robota se zahteva pokazivanje autonomnosti i samoorganizovanosti u prisustvu značajnih neodređenosti u sistemu i radnoj okolini. Neophodna je sposobnost učenja ali i generalizacije stečenog znanja, čime se povećava robusnost upravljačkog sistema. U tom smislu neuronske mreže, sa svojim dobrim osobinama, mogu biti od velike koristi.

Pomenuto je nekoliko kompanija (Neurogen, *Neural Systems*, *Hecht Nielsen Neurocomputers*) koje projektuju neuročipove, prototipove neuro robota ili razvijaju nove metode obučavanja. Naveden je znatan broj mogućih primena NM u robotici: sistemi vizije, procesiranje sa taktilnih senzora, upravljanje robotskom hvataljkom, upravljanje kretanjem, analiza situacije, rešenje inverznog kinematičkog problema, koordinacija sa senzorima, učenje vizualno-motorne koordinacije itd.

Postoje tri nivoa upravljanja robotima. Strategijski podrazumeva planiranje radnog zadatka sa robotskom inteligencijom: prepoznavanje senzornih informacija, prepoznavanje dubine u sistemima vizije. Taktički pretpostavlja planiranje optimalne trajektorije uz zaobilaženje prepreka na osnovu ograničenih informacija. Izvršni ima u vidu upravljanje robotskim kretanjem. Nemamo, na žalost, prostora za temeljniji prikaz ovog zanimljivog i opsežnog rada. Posle prikazanih iskustava u ovoj problematici zaključeno je da se pred NM postavljaju brojni zahtevi koji bi omogućili veći prodor NM u robotiku. To su pitanja optimalne strukture mreže, konvergencije i povećanja brzine učenja itd.

Mašinsko učenje neanalitičke kontrole pokreta Dejana Popovića bavi se mogućom kontrolom pokreta ekstremiteta ljudi koji su pretrpeli oštećenje nervnog sistema. Ovo elokventno predavanje predstavlja taj problem kao krajnje različit od problema robotskog upravljanja. Naime, jednostavni čovekovi pokreti imaju desetine stepeni slobode, a pri tom se aktivira na stotine mišića (aktuatora). Danas raspoloživa tehnika, uključujući i NM, nalazi na velike teškoće pred ovako složenim zadacima upravljanja gde je potrebno sintetizovati prirodne i uobičajeno brze pokrete ruku ili nogu. Ovaj problem je utoliko teži što je veća atrofija mišića kao prirodna posledica zamrlih nerava i nekretanja.

S druge strane, kada je u pitanju sistem koji treba da omogući hod, obično prihvatljiva greška od 3% je u ovom slučaju katastrofalna jer znači pad na svakih 30 koraka! Kao mogući uzrok ovim preprekama na duhovit način su analizirane dijametralne razlike metoda učenja svojstvenih čoveku i kompjuteru. Naime, čovek počinje da uči na osnovu posmatranja i otkrivanja, zatim uči iz primera, pa onda na osnovu analogije, zatim uči ono što mu drugi kaže i završava trulim učenjem (jer ne zna šta drugo da radi). Računarski pristup je upravo obrnut! Na kraju je predstavljeno i jedno rešenje koje omogućuje hod, primenjeno u praksi. Zaključeno je da ova problematika očekuje veoma mnogo od razvoja novih tehnika i bržih kompjutera.

I time završavamo prvi deo ovog prikaza. Za sledeći broj ostavljamo radove nešto više posvećene elektrotehnici i medicini. ■

Autora možete kontaktirati na e-mail adresu: isajic@sezam.co.yu

EDI - mesto i uloga

U poslednje vreme dosta se govori o informacionim magistralama, otvorenim sistemima, EDI, UN/EDIFACT standardima i drugim konceptima koji su razvijeni tokom poslednjih decenija i rezultat su primene računarskih i komunikacionih tehnologija. Šta navedeni pojmovi znače?

mr Milan Milošević

U poslovnoj praksi sve se češće koristi pojam EDI elektronska razmena podataka (*Electronic Data Interchange*). Postoji više definicija EDI od kojih se najčešće koristi sledeća: EDI je razmena strukturiranih komercijalnih podataka između računara zasebnih firmi izvršena bez manuelne intervencije elektronskim putem posredstvom standardizovanih poruka koje zamenjuju tradicionalne papirne komercijalne dokumente. Ova definicija u sebi sadrži sledeće ključne elemente:

- EDI je razmena podataka
- EDI je razmena strukturiranih komercijalnih podataka
- EDI je razmena podataka između računara zasebnih firmi
- EDI je razmena podataka između računara bez manuelne intervencije
- EDI je razmena elektronskim putem
- EDI je razmena posredstvom standardizovanih poruka.

Ovo je definicija opravdana sa tehničkog stanovišta. Sa stanovišta korisnika značajnije je posmatrati genezu ekonomskih disciplina koja daje odgovor na mesto i ulogu EDI.

U vreme oskudice materijalnih dobara kojima se podmiruju osnovne ljudske potrebe, odnosno u uslovima u kojima ponuda robe nije mogla podmiriti postojeću tražnju, ključno pitanje privrednog razvoja je bilo KAKO PROIZVESTI. Traženje odgovora na ovo pitanje vodilo je preko nastanka i razvoja industrijskog inženjeringa. Mnogobrojne metode koje su u okviru njega razvijene bile su usmerene na unapređenje produktivnosti rada, odnosno imale su za cilj povećanje obima proizvodnje sa istim ili čak smanjenim ulaganjima živog i minulog rada.

Usavršavanje radnih operacija i racionalizacija procesa proizvodnje, a naročito automatizacija tehnoloških procesa, rešavaju problem obima ponude ali ne dovode do nestanka poslovnih problema. Poslovni problemi se početkom tridesetih godina ovog stoleća pomeraju iz sfere proizvodnje u sferu robnog prometa. Ključno poslovno pitanje više nije kako proizvesti već KAKO PRODATI?

Pomeranje težišta poslovne problematike daje impuls razvoju marketinga, ekonomske discipline orijentisane prema kupcu, odnosno prema potrošaču i njegovim potrebama.

Razvoj i praktična primena istraživanja tržišta, privredne propagande, psihologije potrošača i drugih disciplina u okvirima marketinga, u uslovima porasta platežno sposobne tražnje, rešava problem gria u sferi robnog prometa. Šire se postojeći i pronalaze novi kanali prodaje, stvaraju i razvijaju nove potrebe kod potrošača, smanjuju zalihe gotove robe, omogućuje se uticaj potrošača na projektovanje i razvoj novog proizvoda, a sve ovo se ogleda u povećanju ekonomske efikasnosti preduzeća i ubrzavanju razvoja zemalja sa tržišno orijentisanom privredom.

U konkurentskoj borbi na tržištu osnovno oružje jeste unapređenje kvaliteta uz istu prodajnu cenu, odnosno sniženje prodajne cene uz održavanje visokog kvaliteta. U uslovima u kojima je kvalitet proizvoda dostigao zavidan nivo, put ka poboljšanju pozicije na tržištu vodi logično preko daljeg snižavanja cene koštanja, odnosno daljeg sniženja prodajne cene proizvoda. Nastojanje da se uspešnost poslovanja dalje unapredi u ovom smislu vodi ka novom pomeranju težišta poslovne problematike. Pošto su u fazi unapređenja produktivnosti rada u centru pažnje ekonomista bili ne samo troškovi rada, već i troškovi sirovina i materijala, u oblasti daljeg smanjenja direktnih troškova poslovanja nisu se više mogle ostvariti značajne uštede.

Velikoserijska i masovna proizvodnja, odnosno krupni nefleksibilni proizvodni sistemi, uslovili su veliki porast angažovanih sredstava u svim fazama ciklusa reprodukcije, tako da su, uprkos smanjenju direktnih utrošaka rada i materijala, ukupni troškovi po jedinici proizvoda bili znatno iznad objektivno uslovljenih.

Ključno poslovno pitanje u savremenoj privredi više nije kako prodati, već KAKO SMANJITI SUMU ANGAŽOVANIH SREDSTAVA po jedinici ostvarene dobiti. Odgovor na ovo pitanje treba tražiti u oblasti ubrzanja cirkulacije angažovanih sredstava u svim fazama ciklusa reprodukcije. Iskustva razvijenih zemalja, a posebno Japana, pokazuju da u oblasti racionalizacije angažovanih sredstava leže velike rezerve čije aktiviranje predstavlja jedan od najznačajnijih instrumenata za unapređenje uspešnosti poslovanja.

Naučno uopštavanje svih praktičnih napora na ubrzanju cirkulacije poslovnih sredstava, s jedne strane, i razvoj vojne logistike s druge strane, doveli su do nastanka nove ekonomske discipline - poslovne logistike.

Tu, u pojavi poslovne logistike, treba tražiti mesto EDI. EDI je sa stanovišta poslovne logistike jedan, mada ne i jedini, instrument rešavanja problema kojima se poslovna logistika bavi. Kao nauka, poslovna logistika predstavlja ekonomsku disciplinu koja proučava tokove i transformacije ekonomskih sadržaja u okviru preduzeća (poslovnog sistema). Ona time stvara naučno uopštena rešenja (modele, metode i tehnike) za prevazilaženje prostorne i vremenske dimenzije ciklusa reprodukcije.

Poslovna logistika kao poslovna funkcija predstavlja skup svih aktivnosti čija je svrha povećanje propulzivnosti i ekonomske efikasnosti poslovnog sistema.

Poslovna logistika kao koncept može da egzistira nezavisno od EDI, ali tek primenom EDI ona daje efekte koji su razlog intenzivnog uvođenja EDI u poslovanju. Nije slučajno to što je EDI prvo uveden u velikim firmama koje su imale veliku sumu angažovanih sredstava i koje su intenzivno tražile odgovor na pitanje "Kako smanjiti sumu angažovanih sredstava za nesmetano odvijanje poslovnih operacija". Svaka tehnologija ima svoje limite. Poslovna logistika bazirana na razmeni poslovnih poruka na papiru i poštom je jedno vreme davala rezultate. Razvoj tehnologije je omogućio primenu teleksa, ali je i on tehnološki zastareo pojavom telefaksa. Pojava telefaksa znatno je ubrzala razmenu poslovnih poruka, ali time osnovni problemi nisu otklonjeni. Poboljšanja perfomansi su bila prisutna ali ne u onoj meri koja bi zadovoljila savremenu firmu.

Sedamdesetih godina ovog veka pojavio se koncept elektronske razmene podataka. Vrlo brzo su uprave firmi uvidele značaj primene tog koncepta za rešavanje njihovih poslovnih problema, uložile su novac, vreme i talenat svojih najboljih stručnjaka i - dobili smo koncept EDI. EDI nije tehnologija ili njeno unapređenje. EDI je pre svega razuman poslovni izbor.

Poslovna praksa kaže da primenom EDI imamo barem sledeće efekte:

- smanjuju se troškovi obrade papirne dokumentacije (papir, osoblje i vreme)

- smanjuje se obim ljudskih grešaka u obradi podataka
- brža i bolja informisanost i dostava informacija,
- efikasnije upravljanje novčanim sredstvima
- mogućnost poboljšanja upravljanja zalihama
- poboljšano upravljanje transportom i distribucijom
- pojednostavljenje i ubrzanje carinskih procedura
- povećanje produktivnosti
- prevazilaženje jezičkih barijera i problema rada u različitim vremenskim zonama
- i drugo

Države koje žele da poboljšaju poslovne performanse svoje privrede intenzivno ulažu u razvoj informacione tehnologije a posebno u razvoj EDI Rastom broja organizacija koje imaju uveden EDI ekonomski efekt su veći Idealno je da sve firme koriste EDI što je cilj koji će se postići tek u narednom veku

Amerika gradi nacionalnu informacionu strukturu Evropa razvija koncept informatičkog društva Isto važi za druge regione i ekonomske grupacije Ujedinjene nacije razvijaju koncept globalne informatičke infrastrukture Ovi koncepti nisu međusobno konkurentni Naprotiv!

Nove tehnologije i novi poslovni postupci zahtevaju da ponovo tražimo odgovor na suštinska pitanja privrednog i društvenog razvoja u celini Nacionalni razvojni prioritet razvijenih zemalja je razvoj i primena informacionih i komunikacionih tehnologija i sistema Nacionalni razvojni prioritet Jugoslavije u nastupajućim godinama je razvoj nacionalne informacione infrastrukture kao kičme ukupnog privrednog i društvenog razvoja ■

Autora možete kontaktirati na sledeće e mail adrese SMTP mil mil@fon fon bg ac yu i WWW http //fon fon bg ac yu/ ~milmil/

3. YUEDI konferencija

Jugoslovenska asocijacija za elektronsku razmenu podataka u saradnji sa Saveznim ministarstvom za razvoj nauku i životnu sredinu Saveznim zavodom za standardizaciju Službom za platni promet Jugoslavije Privrednom komorom Jugoslavije Privrednom komorom Beograda Jugoslovenskom asocijacijom za numeraciju artikala i uz medijsko sponzorstvo našeg časopisa organizuje 3 YUEDI konferenciju 12 13 decembra 1995 godine Konferencija će se odvijati u Privrednoj komori Jugoslavije Terazije 23 u Beogradu Tema Konferencije će biti Elektronska razmena podataka i elektronska trgovina u svetu i Jugoslaviji - stanje i perspektive Treća YUEDI konferencija sledeći tradiciju prethodnih dveju, treba da obezbedi kontinuitet i korak napred u razvoju i primeni EDI-ja

Cilj Treće YUEDI konferencije je da se pruži celoviti pogled na razvoj elektronske razmene podataka naročito elektronske trgovine u svetu i Jugoslaviji Imajući u vidu činjenicu da sve razvijene zemlje sveta izuzetno velika sredstva ulažu u razvoj EDI/EC (*Electronic Data Interchange/Electronic Commerce*) značaj EDI/EC kao odlučujućeg faktora konkurentnosti u međunarodnoj podeli rada i shvatajući da je razvoj i primena EDI/EC razvojni prioritet Jugoslavije YUEDI asocijacija organizuje Treću YUEDI konferenciju koja treba da da odgovor na pitanja gde se EDI/EC danas nalazi i koji su očekivani pravci njegovog razvoja

Konferencija treba da omogući kvalitativni pomak u primeni koncepta elektronskog privređivanja u Jugoslaviji imajući u vidu značaj elektronske razmene podataka elektronske trgovine Skup je namenjen svima onima koji žele da se upoznaju sa konceptom elektronskog privređivanja u svetu i Jugoslaviji i koji žele da ga primene za profitabilno poslovanje organizacija u kojima rade to su menadžeri na svim nivoima upravljačke hijerarhije ekonomisti inženjeri i drugi profili stručnjaka. Ciljna publika će sem iz privrednih i vanprivrednih organizacija biti i iz državnih i finansijskih ustanova U njoj će se osim ljudi već upoznatih sa konceptom EDI (Electronic Data Interchange) elektronske trgovine i elektronskog privređivanja u celini naći i oni koji žele da steknu nova znanja i da ih primene

U toku održavanja konferencije predviđeno je organizovanje izložbe hardvera i softvera kao i multimedijalne prezentacije hardverskih i softverskih rešenja za primenu u EDI/EC

Pre u toku i posle Konferencije planirano je da se organizuje više okruglih stolova i panel diskusija na kojima bi se pojedina pitanja detaljno obradila

Zbornik radova biće distribuiran i na disketama a izdanje u hipertekstu biće instalirano na WWW serverima u Jugoslaviji i svetu

U toku 3 YUEDI konferencije predviđeno je više multimedijalnih prezentacija iz domena elektronske razmene podataka elektronske trgovine kao i prezentacija firmi koje se bave proizvodnjom softvera i hardvera za EDI ■

Detaljne informacije i prijavljivanje
YUEDI asocijacija, 11000 Beograd
Kneza Miloša 20 Jugoslavija
Telefoni
381 11-324-90 28 +381 11 681 999/197
Fax +381-11-235 10 36
SMTP yuedi@osmeh fon bg ac yu
X 400 /S=YUEDI/A=BEOGRAD400
/P=PUBLIC/C=YU/

Oglašivač	Adresa	Telefon	Strana	Br. u servisu
AB soft	Kneza Miloša 82 11000 Beograd	(011) 656-857 646-692	44	15
ADA computers	YU biznis centar Bul. Lenjina 10 11070 N. Beograd	(011) 145 140 145-192 137-397	49	31
Adacom computers	Čika Ljubina 12 11000 Beograd	(011) 337 367, 629-233 341-496	43	
APP systems	Jovana Ristića 6 11000 Beograd	(011) 647-190	117	06
ASYS computers	Bulevar Oslobođenja 59 21000 Novi Sad	(021) 623 928	53	
Beogradski sajam			40	
Biosfera	Dom Omladine Makedonska 22/6, 11000 Beograd	(011) 3229 109 3224-378 3248-208	3K	03
Bit computers	Narodnog Fronta 53 21000 Novi Sad	(021) 369 191 365 222 (lok. 175 179)	13	01
BM	Vojvode Milenka 9 11000 Beograd	(011) 644 398 645-227	126	38
Boleco	Sremskih odreda 14/20 11070 Novi Beograd	(011) 674-242 459-557	97	
BS Procesor	Hadži Nikole Živkovića 2, 11000 Beograd	(011)180-739 183 832 632 162, 632-882	54	11
CET	Terazije 13/VI 11000 Beograd	(011) 343 043 631-789	8	18
Compex	Mutapova 41 11000 Beograd	(011) 432 - 495 432 806	36	34
Computer dream	Nemanjina 4 11000 Beograd	(011) 3226 303 3226-323 3224-221	72	
Comtec	Panske Komune 14 PO box 39 21000 Novi Sad	(021) 411-533 413 918	18 19	10
ComTrade	Genex apartmani 11070 Novi Beograd	(011) 222-4151 222-2652, 222-4139	4K	13
Delfin data	Gavrila Principa 44a 11000 Beograd	(011) 688-797	30	
El S graf	Bul. Veljka Vlahovića 18000 Niš	(018) 25 720 347-089	35	37
Electronic Design	Maršala Tolbuhina bb Pejton 11000 Beograd	(011) 450-480 444-7459	59	
Elegra	Ivana Milutinovića 16a 11000 Beograd	(011) 434-012 452-645, 444-9758	42	
Eurokomerc	Terazije 41/2 11000 Beograd	(011) 331-450, 334-129	2K	22
IMTEL	Bulevar Lenjina 165b 11070 Novi Beograd	(011) 135-420, 142-164 134-516 222 4078	53	17
Institut Vinča centar	Beogradanka Masarikova 5/XI 11000 Beograd	(011) 683-390, 682 486	80	08
ISC	Lomina 36 11000 Beograd	(011) 659-506 681 949	78	05
Jitex	Oblakovska 57a 11000 Beograd	(011) 647-141 650-777 767 832	72	14
KMA S	Ubička 24 11000 Beograd	(011) 663-182 664 934 664-934	66	09
Kompani	Cara Lazara 16 11000 Beograd	(011) 627 525 627-413 626-741 624-795	113	
Logic Design	Tadeuša Košćuška 34 11000 Beograd	(011) 180 611	119	
Matrix	Kursulina 15/II 11000 Beograd	(011) 436-094 444-1867 435 915	117	
Megabit	Bulevar umetnosti 27 11070 Novi Beograd	(011) 2222 693	96	
Meridian Trade	Zmaj Jovina 4 11000 Beograd	(011) 620-374 632-996 626 576	28	07
Microsys	Bulevar Crvene Armije 40 11000 Beograd	(011) 444-9730 444-9099 444-3675	31	16
Mikro knjiga	Petra Martinovića 6 11000 Beograd	(011) 2540-544 2540-545 542-516	48	19
MZ computers	Dubljanska 70 11000 Novi Beograd	(011) 434-812, 450-471	65	
Olymp elektronic	Jovana Đaje 10 11000 Beograd	(011) 400-477 410 240	77	12
Orka	Zmaja od Noćaja 3a 11000 Beograd	(011) 3281 483 639 409	66	
Osa	Đunđeva 6 11000 Beograd	(011) 414-822 414 134 410-762, 410-896	77	
OSA računarski inženjering	Narodnog Fronta 56, 11000 Beograd	(011) 681-743 644 058 681-199, 644-567	27	20
OTC computers	Sava Centar M. Popovića 9 11070 Novi Beograd	(011) 139-840 (lok. 684 694) 2222-355	3	21
PC servis			87	25
Penhard inženjering	Ivana Milutinovića 24 11000 Beograd	(011) 436-019 432-319 432 383 435-513	52	04
Ribbon	Vele Nigrinove16 11000 Beograd	(011) 402 910 648-485 423-171	87	
SAGA	Sava Centar Milentija Popovića 9 11000 Beograd	(011) 222 3579 147-182 455-785 605-578	6	36
Spaschex	Todora Jovanovića - Toze 1 21000 Novi Sad	(021) 393 592	130	
Spectrum	7. jula 3 18000 Niš	(018) 24-027	126	
Sprint	Miloša Bajića 1/I 21000 Novi Sad	(021) 623 901 623 717	22	23
Stil computers	Njegoševa 42 11000 Beograd	(011) 444-1565 451 982	14	
Vidra info	Njegoševa 54 11000 Beograd	(011) 444 9456 444 9234, 444-7541	40	30

Da biste dobili besplatni reklamni materijal sa dodatnim informacijama od oglašivača, potrebno je da pošaljete priloženi kupon. Molimo vas da lične podatke ispišete čitkim slovima i da obavezno popunite upitnik koji ima namenu da informiše oglašivače o profilu i oblastima interesovanja potencijalnih kupaca. Zatim isecite i presavijte kupon po naznačenim linijama i pričvrstite spojeve lepljivom trakom. Kuponi sa nepotpunim odgovorima neće biti obrađivani. Priložen je i kupon kojim možete zatražiti pretplatu na naš časopis, nakon čega ćemo vam poslati uplatnicu. Poštarina je unapred plaćena.

Unix servis je rubrika koja donosi *Bourne/Korn shell skript* programe, C programe i savete za lakše i efikasnije korišćenje i održavanje *Unix* mašina.

Vaše priloge možete poslati na redakciju časopisa "Računari", Bul. Vojvode Mišića 17/III, 11000 Beograd, sa naznakom "za UNIX SERVIS". Elektronskom poštom prilog možete poslati na:
unix-servis@galeb.etf.bg.ac.yu

Uputstvo za pisanje priloga možete dobiti automatski ako pošaljete poruku na:
vodic-unix-servis@galeb.etf.bg.ac.yu

Ako vaš prilog bude objavljen biće vam poslato 50 dinara u roku od 30 dana po objavljivanju.

Za materijal objavljen u Unix Servisu ne daje se nikakva garancija. Svako korišćenje materijala objavljenog u Unix Servisu na rizik korisnika.

Koliko god vam Unix činio mnoge stvari lakšim, pak većina ljudi mora dosta stvari raditi i na DOS mašinama. Ko jednom oseti mogućnosti koje daju skriptovi na Unix-u, teško se miri sa slabašnim DOS-ovim interpretatorom komandne linije i njegovim .BAT skriptovima. Tokom vremena bilo je i komercijalnih paketa koji su DOS-u davali Unix interpretator komandne linije, no ovde govorim o deljenom (*shareware*) paketu firme *Data Logic Limited*, koji sadrži kompletan *Korn shell* za DOS i OS/2. Za DOS postoje dve varijante: jedna je obična 16-bitna, a druga 32-bitna sa DOS/4GW DOS ekstenderom. Za OS/2 postoje čak tri verzije: 16-bitna, 32-bitna i 32-bitna sa podrškom za EMX podsistem.

Kompletna arhiva sa svim ovim verzijama velika je nekih 820KB, ali je sam izvršni program velik svega 190KB, dok DOS4GW zauzima oko 240KB. Sve se to nalazi u arhivi *ms_sh23b.zip* koja bi u vreme kad čitate ovaj tekst trebalo da se nalazi na barem jednom od javnih ftp čvorova, kao i nekoliko arhiva sa raznim Unix-ovskim komandama koje možete koristiti kako uz ovu ljusku, tako i samostalno iz command.com-a. Za one koji nemaju vremena za instaliranje i podešavanje, tu je i minimalna varijanta u arhivi sh.zip, koja sadrži nešto stariju 16-bitnu verziju programa (2.0; tekuća je 2.3), tridesetak osnovnih *Unix*-ovskih komandi i sed.exe koji radi sve što i *Unix*-ov sed. Na vama je da po želji dodate awk, perl i druge programe koji su vam potrebni.

Zasto Korn shell?

Korn shell je kompatibilan unazad sa *Bourne shell*-om, ali ima bitno poboljšan interaktivni rad na komandnoj liniji, kao i mogućnost pravljenja funkcija unutar skripta. Brojni korisnici se, kad god je to moguće, opredeljuju za *bash*, koji je GNU-ova varijanta *Bourne/Korn shell*-a. Zapravo, danas još samo poneki proizvođači *Unix*-a insistiraju na starom *Bourne shell*-u. Obično ugrađuju proširenja koja donosi *Korn shell* u sam *Bourne shell*, tako da se praktično gubi razlika.

Ukoliko se opredelite da sami instalirate *Korn shell*, preimenujte izabrani .exe u sh.exe, jer svi skriptovi koje budete prenosili sa *Unix*-a očekuju to ime. Samo instaliranje svodi se na kopiranje i podešavanje par osnovnih inicijalizacionih spisa. Ukoliko želite 32-bitnu verziju, tome dodajte i podešavanje DOS4GW-a, no to vam obično nije potrebno, sem ako baš imate jako mnogo posla da uradite, no tada je ipak bolje držati se čistog *Unix*-a.

Posle raspakovanja treba u DOS-ov PATH uneti direktorij x:/BIN, a bilo bi dobro i x:/ETC (gde je x ime particije na kojoj se sve nalazi; preporučljivo je da to bude C:). Ljusku uvek startujte sa sh -0, što će obezbediti ispravno učitavanje inicijalizacionih spisa. Bez problema možete raditi i iz *MS Windows*-a, kroz DOS prozor, naravno.

U većini slučajeva *Korn shell* nećete koristiti za stalni rad (mada je i to sasvim izvodljivo), već samo za pisanje, proveravanje i izvršavanje skriptova. Možete čak pisati skriptove iz svog omiljenog *Windows* editora, a onda ih iz DOS prozora proveravati i izvršavati. Za maksimalnu brzinu je, naravno, preporučljivo raditi bez opterećenja *Windows*-a, no ako se mali gubitak brzine ostavi po strani, program je izuzetno stabilan i radi već više od godinu dana bez ijednog pada *Windows*-a ili kakvog drugog problema (što ne znači da dovoljno uporan ili dovoljno nesrećan korisnik ne može doći i do toga).

Za brzo snalazenje

Ovo je upravo zgodna prilika da oni koji žele da počnu da rade sa *Unix*-ovskom ljuskom dobiju mali uvod i nešto objašnjenja za početak. Rad sa *Unix* interpretatorom komandne linije mnogo će vam ličiti na rad u DOS-u, jer je DOS u svom nastajanju dosta stvari "pozajmio" od *Unix*-a.

Ono što znate iz DOS-a je **cd** naredba, ali pazite: *Unix* koristi kosu crtu ("/"), a ne obratnu kosu crtu ("\") na koju ste navikli. Naredbe dir nema, za listu imena spisa i direktorija koristite **ls**, pri čemu su najkorisnije kombinacije sa opcijama **ls -l** i **ls -CF**. Umesto **type <ime_spisa>** koristite **cat <ime spisa> | more**.

Za kopiranje spisa koristite naredbu **cp**, a za premeštanje **mv**. Direktoriji se stvaraju sa **mkdir**, a uklanjaju sa **rmdir**, ako su prazni. Za uklanjanje celog stabla direktorija sa svim spisima koristi se **rm -r**. Kad želite da vidite u kojem ste direktoriju, izvršite **pwd**.

Od svih do sada pomenutih komandi samo su **cd** i **pwd** interne, a ostale su eksterne, što znači da morate imati poseban .exe koji čini tu komandu (u sh.zip ih već imate u bin/ direktoriju). Za sve interne komande dokumentacija je u spisu sh.man, no on je predviđen za čitanje iz shell-a, pa mnogo koristi znak unazad (*backspace*, BS - \x08, \010) za podvlačenje i pojačavanje slova. Ako to želite da očistite, iskoristite bilo koji malo bolji tekst editor koji ima džokere u pretraživanju i zamenite znak pre BS-a i sam BS praznim stringom. To možete uraditi i sed-om kao:

```
sed "s/.\08//g" sh.man > sh.txt
```

Takvim čišćenjem uvek možete brzo dobiti običan tekst za čitanje u DOS-u, no tu postoji još jedna sitnica. Na *Unix*-u je znak za kraj reda samo LF, dok je na DOS-u to par CR, LF. Zato su u sh.zip uključeni i u2d.com (*Unix to Dos*) i d2u.com (*Dos to Unix*) za brzo prebacivanje formata. Kao i svaki drugi DOS program, i njih možete slobodno izvršavati i iz *Korn shell*-a.

Dokumentacija u sh.man je formatirana na 57 stranica običnog iglićnog štampača. Uz dobar prelom (dve kolone, smanjen razmak između redova i veličina slova), međutim, može se odštampati na laserskom štampaču i na nekih 12-15 stranica.

Na raspolaganju u radu sa *Korn shell*-om uvek imate i prethodno izvršene komande koristeći obične kursore, ako je sve dobro podešeno, a za one koji su navikli, postoje i vi i *emacs* režimi rada. Internom naredbom **history** možete na ekranu prikazati sve ranije izvršene komande. Pošto se ove komande smeštaju u spis na disku, imate sačuvane i komande koje ste izvršavali pre nekoliko dana.

Redirekcija podataka ostvaruje se poznatim < > >> operatorima, a imate na raspolaganju i cev ("|"). No, pošto ipak niste na *Unix*-u, redirekcija se ostvaruje kroz privremene spise, a ne uporednim izvršavanjem programa, tako da dobijate samo eleganciju rada, ali ne i dodatnu brzinu.

Programiranje

Programiranje, tj. pisanje i korišćenje skriptova je glavni razlog zbog kojeg uopšte instalirate *Korn shell* na DOS mašinu. Naredbe navedene u skriptu se jednostavno izvršavaju jedna za drugom. Možete staviti više naredbi u jednu liniju, ako svaku završite tačkom-zarezom (";"). Kontrolne strukture su vrlo slične onima u C-u, tako da za one koji znaju makar malo C problema ne bi trebalo da bude.

Osnovna petlja se ostvaruje sa:

```
while uslovi
do
   naredbe
done
```

i izvršava se sve dok poslednja naredba u listi **uslovi** daje kao rezultat 0 (što znači uspešan završetak ili vrednost "tačno" - malo različito od onoga što očekujete u C-u), a u svakom prolazu izvršava sve naredne iz liste **naredbe**. Umesto **while** može se koristiti **until**, čime se negira uslov, pa se petlja izvršava sve dok dobija vrednost različitu od nule (što označava grešku u izvršavanju ili "netačno").

LISTING 1
Logički testovi

```
-a file spis postoji
-b file spis file je blok specijalni
-c file spis file je znakovni specijalni
-d file direktorij file postoji
-r file spis postoji i obican je (nije specijalan)
-g file spis postoji i ima SETGID bit postavljen
-h file spis postoji i simbolicki je link
-k file spis postoji i ima sticky bit postavljen (za
DOS - ARCHIVED bit)
-n string       duzina stringa nije nula
-o option       opcija je postavljena
-p file spis postoji i tip mu je FIFO ili cev (pipe)
-r file spis postoji i moze se citati
-s file spis postoji i velicina mu je veca od nule
-t [ fildes ]   otvoreni spis sa opisnim brojem fi-
ledes je terminalski uredjaj
-u file spis postoji i ima SETGID bit postavljen
-w file spis postoji, moze se citati i moze se u
njega pisati
-x file spis postoji i izvršan je, ako je dir., moze
se pretrazivati
-z string       duzina stringa je nula
-L file spis postoji i simbolicki je link
-O file spis postoji i vlasnik mu je efektivni kori-
snicki id (za DOS - HIDDEN atribut)
-G file spis postoji i vlasnik mu je efektivni grup-
ni id (za DOS - SYSTEM atribut)
-S file spis postoji i tip mu je socket (utikac)
file1 -nt file2 file1 postoji i noviji je od file2
file1 -ot file2 file1 postoji i stariji je od file2
file1 -ef file2 file1 i file2 se odnose na isti spis
s1 = s2 stringovi s1 i s2 su identicni
s1 != s2stringovi s1 i s2 nisu identicni
s1 < s2 string s1 je leksicki manji od s2 (po ASCII
tablici)
s1 < s2 string s1 je leksicki manji od s2 (po ASCII
tablici)
n1 -eg n2       n1 je jednako n2.
n1 -ne n2       n1 nije jednako n2
n1 -lt n2       n1 je manje od n2
n1 -gt n2       n1 je vece od n2
n1 -le n2       n1 je manje ili jednako n2.
n1 -ge n2       n1 je vece ili jednako n2
```

LISTING 2
Logički operatori

```
( expr )grupisanje izraza
! expr  negacija izraza
expr1 && expr2  logicke operacija "i" (AND)
expr1 || expr2  logicka operacija "ili" (OR)
```

LISTING 3
Arhive sa Simtela, za Unix komande pod DOS-om

```
fileutil\
dlcpmv23.zip    [illegible]  920405  BSD Unix-style file
                                     copying/moving utils
patch12.zip     [illegible]  900712  Port of Unix "patch"
                                     (lvl12) program to
                                     MSDOS
sharexe.zip     [illegible]  880221  Unix-like shar/un
                                     shar for DOS
sum.zip         [illegible]  890412  BSD-Unix compatible
                                     checksum utility,
                                     w/source
tar.zip         29496   900823  Unix-compatible TAR
                                archive ma-
                                ker/extractor
toadshr1.zip    25394   890615  DOS Unix-compatible
                                shar/unshar, w/TP
                                5.0 src

sysutil\
ms_sh23b.zip    829214  940829  Korn shell (MS
                                shell) Release 2.3
                                Binary
ms_sh23s.zip    507708  940829  Korn shell (MS
                                shell). Release 2.3.
                                Source
danix201.zip    297993  940204  Collection of 25
                                Unix-like utils for
                                MS-DOS
dantools.zip    117287  870717  Daniel Norton"s
                                Unix-like utility
                                collection..
dosix30a.zip    43541   900705  UNIX-like utils. DF,
                                DU, HEAD, RM,
                                TOUCH, WC
dosnx22a.zip    247578  930829  DOSNIX v2 2a. Unix-
                                style utilities for
                                MS-DOS
dskut111.zip    76122   910630  Unix-like utils
                                chmod/cp/du/find/
                                /mv/page/ls/rm
pgrap207.zip    112241  910920  Unix-like tools
                                (grep,tab,un
                                tab,print,tolower)
picnix31.zip    120054  880526  Unix-like commands
                                for PC's (1 of 3)
picnix32.zip    137006  880526  Unix-like commands
                                for PC's (2 of 3)
picnix33.zip    97706   880526  Unix-like commands
                                for PC's (3 of 3)
porttool.zip    183972  930114  16 Unix-like tools
                                ported to MS-DOS
pschel35.zip    36784   870717  PC-shell: UNIX
                                System V-like csh
                                for MS-DOS
rstlkit1.zip    52128   891014  28 Unix-like tools
                                for MS-DOS, 1 of 2
rstlkit2.zip    209929  880623  28 Unix-like tools
                                for MS-DOS, 2 of 2
uxutl23a.zip    244107  940827  Unix utilities for
                                DOS. Rel 2.3  1of4
uxutl23b.zip    233119  940827  Unix utilities for
                                DOS. Rel 2.3  2of4
uxutl23c.zip    226562  940827  Unix utilities for
                                DOS. Rel 2.3  3of4
uxutl23d.zip    164501  940827  Unix utilities for
                                DOS. Rel 2.3  4of4
uznix120.zip    340087  941018  UZfulnix v1.20
                                Unix-like utilities
                                for MS-DOS
```

Kao uslovi se mogu, dakle, pojaviti bilo kakve komande, ali se najčešće koristi **test expr** ili njen sinonim **[expr]**, gde je **expr** kombinacija logičkih testova koje vidite u listingu 1. Neki testovi su izmenjeni tako da proveravaju atribute specifične za DOS. Svi testovi koji proveravaju stvari koje ne postoje pod DOS-om, vraćaju kod greške koji znači "netačno" (nije nula). Unutar **[expr]** možete kombinovati testove logičkim operatorima. Ako pak koristite **test expr** onda testovi **-a** i **-o** ne postoje, već označavaju logičke operacije AND i OR, a zagrade se moraju štititi obratnom kosom crtom da ih interpretator ne bi shvatio kao graničnike procesa.

Brojačka petlja je samo uslovno brojačka, a izgleda kao

```
for ime in lista
do
    naredbe
done
```

i u njoj u svakom prolazu promenljiva **ime** dobija vrednost sledeće stavke u spisku **lista** (te na taj način prebrojava elemente u spisku). Ako se **for** petlja zada u skraćenom obliku

```
... ime
    naredbe
done
```

onda u svakom prolazu promenljiva **ime** dobija vrednost sledećeg argumenta skripta, dakle kao da se uradi **ime = $1 shift**.
Osnovno grananje ostvaruje se kontrolnom strukturom

```
if uslovi
then
    naredbe
else
    naredbe
fi
```

ili

```
if uslovi
then
    naredbe
elif uslovi
    naredbe
else
    naredbe
fi
```

gde se prelaz u **then** granu vrši ako je prvi uslov tačan, u **elif** ako je prvi uslov netačan ali je sledeći tačan, dok se **else** grana izvršava ako ni jedan test nije tačan. Broj **elif** grana je proizvoljan, a sve sem **if then fi** je opciono, pa je najjednostavnija struktura grananja.

```
if uslovi; then
    naredbe
fi
```

gde je korišćen tačka-zarez da se uštedi jedan red i struktura učini kompaktnijom.

Struktura višestrukog izbora izgleda kao

```
case string in
uzorak)
    naredbe
    ;;
uzorak)
    naredbe
    ;;
*)
    naredbe
    ;;
esac
```

gde su uzorci proizvoljni regularni *izrazi*, a izvršavaju se naredbe u onoj grani čiji uzorak prvi "prepozna" string dat u **case** naredbi.
Sem ovih osnovnih struktura, *Korn shell* raspolaže i mogućnošću stvaranja i pozivanja funkcija, te pravljenja menija za interaktivni izbor podatka iz zadatog spiska.

Kuda dalje?

Sve je ovo samo kratak uvod, toliko da se snađete u prvim koracima ako do sada niste radili sa *Unix* ovim interpretatorom komandne linije. Glavni sledeći korak je čitanje sh man i upoznavanje sa detaljima sintakse, promenljivim, parametrima skripta i svim ostalim elementima važnim za pisanje skriptova. Programiranje u skript jeziku je jednostavnije, lakše od BASIC a i ne traži mnogo računarskog predznanja, ali traži da pročitate dokumentaciju, a to opet podrazumeva da se dovoljno snalazite u engleskom jeziku. Tekst dokumentacije je pisan prilično jednostavno, pa bi i oni koji imaju problema sa engleskim jezikom trebalo da mogu da ga relativno lako prate uz dobar rečnik.

Sledeći korak je pregledanje arhiva sa raznim *Unix* naredbama za DOS, čitanje njihove dokumentacije i izbor naredbi koje su vam najpotrebnije. Ovde nije nužno da pročitate svu dokumentaciju, ali je morate imati pri ruci kad vam zatreba za određenu naredbu. Ako za neku nedostaje uputstvo, uvek se možete poslužiti stranicom uputstva (*man page*) sa nekog *Unix*-a.

Za pisanje dobrih skriptova trebalo bi da poznajete i **sed**, a po mogućstvu i **awk**. Ukoliko se odlučite za **perl**, onda vam čitav *Korn shell* možda neće više ni biti potreban na DOS mašini, pošto je **perl** vrlo samostalan. Tada ćete se, naravno, odreći interaktivnog rada i sve ćete raditi kao kad samo pišete program koji potom izvršavate.

Listing 3 vam daje spisak zanimljivih arhiva koje postoje na čuvenom Simtel čvoru na Internetu, kao i na istoimenom paketu CD-ROM ova. Uz malo sreće, sve ove arhive ćete moći da nađete na barem jednom od javnih beogradskih ftp čvorova, a verovatno i na Sezamu. ■

Autor je nezavisni konsultant za UNIX i hyper mediju, C/C++ programer i tehnički autor iz Beograda. Možete ga kontaktirati na zarko@galeb.etf.bg.ac.yu ili zarko@osmeh.fon.bg.ac.yu

13h

Da kvalitet prikazivanja ne zavisi samo od rezolucije, već i od broja boja, potvrdio je mod 13h, sa 320x200 tačaka i 256 dostupnih boja

Ivan Petrović

Jedna od najvećih prednosti VGA kartice nad njenim prethodnikom EGA karticom je sposobnost istovremenog prikazivanja 256 različitih boja izabranih iz palete od 262144 boje. To je predstavljalo veliki korak napred u razvoju video kartica. Na žalost, jedini mod sa 256 boja, mod sa oznakom **13h** ima rezoluciju 320x200, što je mnogo lošije od 640x480 koliko je imao mod **12h** sa 16 boja. S druge strane, postojao je način da se on vrlo lako koristi za animaciju, što mu je donelo veliku popularnost među proizvođačima igara.

Standardna organizacija

Postavljanje moda vršimo kao i obično preko BIOS video prekida, pa se na tome nećemo duže zadržati. Organizacija video memorije u modovima sa 256 boja mnogo je jednostavnija od modova sa 16 boja: bit-ravni se ne upotrebljavaju, već je svaka tačka predstavljena jednim bajtom, slično tekst modu. Tako svaki bajt nosi vrednost između 0 i 255 i ona predstavlja indeks u DAC tabeli boja.

Pošto jedan bajt opisuje jednu tačku, jedan red se sastoji od 320 bajtova, pa vrlo lako adresiramo željenu tačku:

```
Ofset=y*320+x=EKRANTBL[y]+x,
```

Ako niste pratili prethodne nastavke, ponovićemo ukratko razlog uvođenja tabele **EKRANTBL**: operacija množenja je veoma spora i treba je izbegavati, tako smo umesto množenja sa 320 uveli tabelu EKRANTBL koja sadrži vrednosti 0, 320, 640, ..., 63680 redom; ova tabela može da se generiše prilikom samog startovanja programa ili može da bude ugrađena u sam kod.

Prostim računom dolazimo do veličine video memorije potrebne za ovaj mod: 320*200=64000 bajtova, pa je očigledno da adresiranje svih tačaka možemo da obavimo u jednom segmentu.

```
void PostaviTacku(int x, int y,
int boja)
{
  *(char far
*)MK_FP(0xA000,EKRANTBL[y]+x)=bo-
ja;
}
```

Pošto sve ovde prikazane procedure ne koriste nikakve dodatne promenljive, zgodno ih je prebaciti u makro, odnosno *in-line* funkciju ukoliko programirate u jeziku C++. Analogno prethodnoj, pišemo proceduru za očitavanje tačke:

```
int OcitajTacku(int x, int y)
{
  return *(char far
*)MK_FP(0xA000,EKRANTBL[y]+x);
}
```

Već smo pomenuli da je za ovaj mod potrebno 64 Kb video memorije. Kako standardna VGA kartica sadrži 256 Kb na sebi, postavlja se pitanje:

Kako upotrebiti preostala 192 kb?

Zaista, zašto ne bismo jednostavno uključili dodatne tri strane? Ovo je teoretski moguće, ali gore opisana organizacija to praktično onemogućava. Ipak, ostvarivo je uz par trikova. Upoznajmo se prvo sa tim kako su informacije o tačkama raspodeljene između bit-ravni.

Kao i u tekst modu, adresiranje tačaka preko segmenta A000h je iluzija, napravljena radi lakšeg programiranja. Ono što se zaista dešava je sledeće: o tačkama se brinu četiri bit-ravni, upotrebljavajući tzv. *Chain 4* mod. U ovom modu, donja dva bita ofset adrese određuju broj bit-ravni kojoj se vrednost šalje. Ova dva bita se potom postavljaju na 0 i tek tako dobijena adresa upotrebljava se kao ofset adresa određene bit-ravni. Tako tri bajta ostaju neupotrebljena između svakog iskorišćenog bajta u svakoj bit-ravni.

Pogledajmo kako izgleda prva linija na ekranu. Prva tačka nalazi se u bit-ravni 0, druga u bit-ravni 1, treća u bit-ravni 2, četvrta u bit-ravni 3, i to sve četiri tačke na ofset adresi 0000h. Zatim, peta tačka se nalazi u bit-ravni 0, šesta u bit-ravni 1, sedma u bit-ravni 2, osma u bit-ravni 3, na ofset adresi 0004h. Na isti način raspoređene su i ostale tačke.

Pošto je 64000 bajtova potrebnih za skladištenje 320*200 tačaka, raspoređeno u četiri ravni, samo je 16 Kb od svake bit-ravni zaista iskorišćeno. Međutim, i dalje nema mesta za

Inicijalizacija moda 13h (alternativni način)

```
void PostaviModAlt(void)
{
  union REGS r;

  EKRANSEG=0xA000
  EKRANSIRINA=320
  EKRANBPL=80
  EKRANVISINA=200

  r.x.ax=0x13
  int86(0x10,&r,&r);

  outportb(0x3C4,4);

  outportb(0x3CF,inportb(0x3CF)&239
);
    // Iskljucujemo bit 4
    (neparno/parno mapiranje) Mode
    ...

  outportb(0x3CE,6);

  outportb(0x3CF,inportb(0x3CF)&253
);
    // Iskljucujemo bit 1 (povezi
    neparnog i parnog mapiran
    ...
  ...
  ...ucujemo ...
  ...an/...
  ...stra

  _fmemset((char
*)MK_FP(0xA000,0),0 ...
  // Brisemo sadrzaj ...
ravni

  outportb(0x3D4,0x14);

  outportb(0x3D5,inportb( ...
);
  // Iskljucujemo bit 6 ...
Word mod)
  // Underline Location ...

  outportb(0x3D4,0x17);

  outportb(0x3D5,inportb(0x3 ...
);
  // Ukljucujemo bit 6 (Byte
mod) Mode Control registra
}
```

dodatne strane u video memoriji. Razlog su "rupe" sačinjene od po tri neiskorišćena bajta. Naš zadatak je da pronađemo način da ih eliminišemo, da bismo to postigli potrebno je da reprogramiramo nekoliko VGA registara.

Prvi korak je isključivanje *Chain 4* i neparnog/parnog moda. Na ovaj način aktivira se neka vrsta linearnog adresiranja, što znači da se ofset adrese ne grupišu u bit-ravni prilikom pristupa video memoriji. Drugi korak je obaveštavanje *CRT* kontrolera o datim promenama, tj. eliminisanje "rupa" od po tri bajta; menjamo mod iz *double word* u *byte*, što znači da *word* mod mora takođe da bude onemogućen.

Sada, pošto je samo prvih 16 Kb u svakoj bit-ravni zauzeto, moguće je smestiti dodatne tri strane u video memoriju. Prva ekranska strana počinje na ofset adresi 0000h, druga na 4000h, treća na 8000h i četvrta na C000h, u svakoj od bit-ravni. Ali, stvari postaju malo komplikovanije prilikom adresiranja, pošto sve tačke nisu dostupne u okviru jednog segmenta (A000h).

Alternativna organizacija

Shodno novoj organizaciji video memorije, ofset adresa svake tačke računa se na sledeći način:

```
Ofset=(y*320+x)/4=(y*80)+(x/4)=EKR
ANTBL[y]+(x'' '''2),
```

Broj bit ravni kojoj je potrebno pristupiti, kao što je već napomenuto, nalazi se u donja dva bita X koordinate; upotrebljavamo ih za određivanje položaja tačke u bit ravni upotrebom *Map Mask* registra. Zatim se direktnim upisom u video memoriju određenoj tački dodeljuje boja.

```
void PostaviTackuAlt(int x, int y,
int boja)
{
  outportb(0x3C4,2);
  outportb(0x3C5,1??????(x&3));
  *(char far *)MK_FP(EKRAN-
SEG,EKRANTBL[y]+(x '''  2))=boja;
}
```

Slično, očitavanje tačke vrši se pomoću *Read Map Select* registra.

```
int OcitajTackuAlt(int x, int y)
{
  outportb(0x3CE,4),
  outportb(0x3CF,x&3),
  return *(char far *)MK_FP(EKRAN-
SEG,EKRANTBL[y]+(x '''''2)),
}
```

Kako se se biraju različite strane?

Umesto da budu birane od strane rutine za postavljanje tačke (**PostaviTackuAlt**), ovaj zadatak obavlja posebna procedura (**PostaviStranuAlt**); razlog je naravno dobitak na brzini. Vezu između te dve procedure čini promenljiva **EKRANSEG**, koja uvek sadrži segment adresu aktivne strane. Tako EKRANSEG sadrži A000h ukoliko je aktivna prva strana, A400h za drugu, A800h za treću i AC00h za četvrtu stranu. Takođe je potrebna i rutina za prikazivanje željene strane, koje se vrši menjanjem *Start Adress (High)* registra.

```
void PrikaziStranuAlt(char strana)

{
  outportb(0x3D4,0x0C),
  outportb(0x3D5,strana??????6)

  while(inportb(0x3DA)&8);
  while((inportb(0x3DA)&8)==0),
}
```

Ukoliko ste pažljivo pratili ranije nastavke, sigurno ste primetili da smo ovaj registar već upotrebljavali za efekat skrolovanja.

Autor je direktor agencije "Opsenarijum" koja se bavi izradom multimedijalnog softvera i marketingom. Možete ga kontaktirati na e-mail adresu omega@sezam.co.yu ■

Mala škola HTML-a (I)

Svedoci smo da se *WWW (World Wide Web)* manija proširila i na naše prostore. "Računari" su joj u prethodnim brojevima posvetili nekoliko tekstova. U svima njima najviše pažnje posvećeno je predstavljanju WWW a sada je došlo vreme da se predstavi i sam jezik za pisanje prezentacija - *HTML*. Jezik za pravljenje WWW prezentacija je *Hyper Text Markup Language* (HTML). HTML je za kratko vreme doživeo brojne izmene. Poslednja je verzija 3. od 28. marta 1995. godine.

Osnovna ideja prilikom uvođenja HTML-a bila je da se na jednostavan način predstave hipermedijalne prezentacije koje bi bile nezavisne od platforme, u tome se potpuno uspelo. Brze promene HTML-a uslovljene su sve većim zahtevima za sve složenijim WWW prezentacijama. Danas je pisanje WWW prezentacija ne samo zabava, već i veoma tražen posao na Zapadu, naravno. Česte promene standarda povlače za sobom i česte revizije WWW čitača koji prate promene jezika. Ipak, često WWW čitači ne podržavaju u potpunosti standard jezika.

Kroz seriju tekstova naučićete da napišete svoje WWW prezentacije. Počećemo sa osnovama HTML-a (prikazivanjem, povezivanjem tekstova, slika, zvuka...) pa sve do naprednijih tehnika startovanja programa i skriptova na HTTP serveru, povezivanja sa drugim uslugama sistema i drugo. Najbolji način da naučite trikove pisanja efektnih prezentacija jeste da razgledate već gotove: na YU internetu postoji veliki izbor efektnih prezentacija čiji kod možete videti iz svakog WWW čitača.

Struktura HTML dokumenata

U sledećem primeru upoznaćemo minimalni HTML dokument:

```
<HTML>
<HEAD>
<TITLE>Minimalni HTML dokument</TI
TLE>
</HEAD>
<BODY>
<H1>
Minimalni HTML dokument
</H1>
Dobrodosli u svet WWW!
Ovo je prvi paragraf <P>
Ovo je drugi paragraf <P>
</HTML>
```

- Primer 1

HTML koristi oznake da bi saopštio WWW čitačima kako da prikažu tekst. Gornji primer koristi oznake:

- <HTML> koja ide u paru sa oznakom </HTML> na kraju HTML dokumenta i time ga identifikuje, nije obavezan u upotrebi.
- <HEAD> koja ide u paru sa oznakom </HEAD> i koja označava zaglavlje dokumenta.
- <TITLE> (koja ide u paru sa oznakom </TITLE> na kraju naslova), koja određuje naslov dokumenta. U WWW čitačima (primer: NCSA *Mosaic*) naslov je vidljiv u okviru samog *Mosaic*-a i ne javlja se u samom tekstu dokumenta. Da bismo ga prikazali u samom tekstu moramo da koristimo
- oznaku hedera <H1> (kraj hedera je označen sa </H1>)
- oznaku kraja paragrafa <P>

HTML oznaka sastoji se od matematičkog simbola za poređenje "manje" (<) sa leve strane, sledi neki tekst (nazovimo ga direktivom), zatvara se sa matematičkim simbolom za poređenje "veće" (>). To bi najčešće bila početna HTML oznaka. [illegible] oznake obično uparene, na primer <H1> [illegible] </H1>. Oznaka kojom završavamo (na kraju teksta na koji se uputstvo odnosi) ista je kao početna oznaka, osim što se spred teksta (direktive) nalazi kosa crta (/).

Postoje i izuzeci od ovog pravila. Tako oznaka kraja paragrafa nema svoj par. Ona se postavlja na kraj odeljka i njime se on ističe kao posebna celina. Oznaka </P> ne postoji.

HTML ne razlikuje velika i mala slova. Sa stanovišta HTML savršeno je svejedno da li ćete pisati, na primer <TiTLE> ili <tITlE>. Ipak, uobičajeno je da se poštuje konvencija da se direktive pišu velikim slovima.

Bitna je napomena da sve oznake definisane HTML-om nisu podržane svim WWW čitačima. Ako čitač koji koristite za pregled HTLM dokumenta ne podržava neku oznaku, ista će biti ignorisana.

Očigledno je da ponekad postoji potreba da se u HTML dokumentu koriste znaci za manje i veće. Tada ćemo umesto da navedemo recimo x < 13 napisati x<13, odnosno za x > 13 napisaćemo x>13.

Sa stanovišta HTML upotreba dodatnih razmaka i/ili redova u samom dokumentu ne znači ništa za konačni prikaz tog dokumenta od strane WWW viewer-a. Viewer ignoriše sve dodatne znake razmaka i prazne redove. Da bi se počeo novi odeljak mora se koristiti oznaka kraja paragrafa <P>. Izuzetak je "preformatirani" tekst. Znači, sa stanovišta HTML, sledeći dokument je potpuno identičan našem prvom primeru:

```
<TITLE>Minimalni HTML dokument</TI-
TLE><H1>
Minimalni HTML dokument</H1>
Dobrodosli u svet WWW!
Ovo je prvi paragraf <P>
Ovo je drugi paragraf <P>
```

- Primer 2

Naravno, ovo nije nimalo pregledno, veoma je teško za čitanje. Uobičajeno je da se HTML dokument piše što preglednije: da se paragrafi odvajaju bar jednim praznim redom, da naslov i hederi idu u posebnim linijama.

HTML podržava šest nivoa hedera (numerisanih od 1 do 6), od kojih se 1 prvi koristi. Hederi se obično prikazuju naglašenije od običnog teksta, različito zavisno od nivoa. Standardom HTML se ne propisuje kako viewer treba da prikaže heder - obično se prilikom konfiguracije to može podesiti. Prvi heder u svakom dokumentu mora da bude označen kao <H1>. Opšta sintaksa oznake za hedere je:

```
<Hi>Tekst hedera</Hi>
```

gde i može biti od 1 do 6, određuje nivo hedera.

Komentarom se u HTML dokumentima smatra svaki tekst između oznake "<!--" i "-->":

```
<!-- Primer komentara -->
```

ovakav tekst neće se prikazati prilikom pregleda.

Još jedna značajna napomena: u HTML-u je načelno dozvoljeno ugnježđavanje direktiva, ali ne preklapanje. Sledeći primer nije korektan i rezultovaće greškom:

```
<H1><B>Pogresan primer</H1></B>
```

Povezivanja sa ostalim dokumentima

Osnovna snaga HTML nalazi se u sposobnosti povezivanja dela teksta (ili neke slike) sa drugim doku[illegible] (ili slikom). Taj deo teksta obično je obe[illegible] od strane viewera da bi se naznačilo da tu postoji hipertekst veza.

U samom HTML dokumentu hipertekst veza [illegible]:

```
<A HREF="sledeci html"> Sledeci
tekst za prikaz</A>
```

- Primer 3

Veza se otvara oznakom "<A". Zatim sledi specifikacija dokumenta na koji se ukazuje sa parametrom HREF="sledeci html" iza kojeg zatvaramo oznaku sa znakom "veće" (>). Iza toga se unosi tekst koji će biti obeležen kao hipertekst veza. Sve se završava oznakom kraja veze </A>. Ovo je način povezivanja sa dokumentima na istom mestu.

Ako želimo da povežemo neko mesto u jednom dokumentu sa tačno određenim mestom u drugom, a ne sa njegovim početkom, moramo da menjujemo vezu. U drugom dokumentu, na mesto na koje želimo da se prebacimo iz prvog dokumenta, upisaćemo oznaku ulaza.

```
Ovo je <A NAME="Skok_iz_prvog"> me-
sto ulaza</A> iz prvog dokumenta
```

Kada kreiramo vezu iz prvog dokumenta, nećemo samo upisati ime HTML dokumenta, već i ime ulaza u tom dokumentu koji želimo da nam viewer pokaže.

```
A ovo je <A HREF="dokument_dru-
gi.html#Skok iz_prvog"> veza</A>
```

Ako se sada klikne na reč "veza" u prvom dokumentu, viewer će nam prikazati dokument počev od reči "mesto ulaza" u drugom dokumentu.

Ukoliko želimo da sa jednog mesta ukažemo na drugo mesto u **istom** dokumentu, ceo postupak će biti isti, s tim što ćemo na mestu imena HTML dokumenta upisati isti taj dokument.

URL

URL (Uniform Resource Locator) odnosi se na format korišćen od strane WWW dokumenta da se locira drugi fajl. URL daje tip usluge koja se koristi (na primer FTP) i putanju do fajla koji se traži. Opšti format je:

```
tip //host domen[ port]/puta-
nja/ime_fajla
```

gde je tip jedna od sledećih usluga:

- *file* fajl na lokalnom računarskom sistemu ili fajl na anonimnom FTP serveru
- *http* fajl na WWW serveru
- *gopher* fajl na Gopher serveru
- *WAIS* fajl na WAIS serveru
- *telnet* pristup preko telneta udaljenim računarima,
- *news* čitanje news grupa.

Primer za vezu ka spoljnom HTML dokumentu bi bio:

```
<A HREF="http //zmaj.etf.bg ac yu
/WWW/ ETF_Beograd.html"> O ETF-u!
</A>
```

Ovaj primer u našem HTML dokumentu učinio bi tekst "O ETF-u!" hipertekst vezom ka naznačenom HTML dokumentu.

A šta je sa slikama?

Većina WWW čitača, grafički orijentisanih, u stanju je da prikaže slike u GIF u kao osnovnom formatu. Sa da postoji i podrška za JPG, ali neki čitači to podržavaju samo uz poziv eksternog programa. Za prikaz svake slike potrebno je vreme (pre svega za prenošenje slike preko mreže, što zna da bude itekako primetno ako vam je brzina komunikacije mala) i usporava prikaz dokumenta. Ako se ista slika prikaže nekoliko puta u istom dokumentu, degradacija performansi je mnogo manja nego kada bi se prikazalo više slika po jedanput.

Izgled vaše prezentacije može se značajno poboljšati uključivanjem manjih slika i ikona u nju. Za dobar izgled prezentacije upotreba efektnih slika je presudna. Sliku ćemo u prezentaciju uključiti sa znakom manje, direktivom IMG (<IMG) koja može da ima sledeće atribute:

- SRC – njegova vrednost treba da bude URL slike koja se prikazuje
- ALIGN – može uzeti vrednost TOP, MIDDLE i BOTTOM, što određuje gde će se tekst poravnati sa slikom
- ALT – opcionaln tekst koji se prikazuje ako WWW čitač nije u stanju da prikaže sliku

```
< MG SRC="slika.gif">
```

Primer za prikaz slike

Tako je moguće pozvati sliku iz dokumenta. To je zgodno kada se ne želi opterećivati osnovni dokument preteranim usporavanjem zbog prikaza slika, već se daje mogućnost da se klikom na vezu ista prikaže.

```
<A HREF='slika.gif"> Veza ka sli-
ci </A>
```

Liste

HTML podržava sledeće tipove lista: nabrajanja, navođenja i opisne liste. Za odvajanje članova lista nije potrebna oznaka kraja paragrafa. Njih odvaja oznaka člana liste.

- **Liste navođenja**

Počinju oznakom otvaranja liste <UL>. Svaki član liste počinje sa oznakom <LI>. Nema oznake kraja člana liste. Listu zatvaramo oznakom zatvaranja liste </UL>.

Primer za listu navođenja

```
<UL>
<LI>ETF
<LI>ostali
</UL>
```

Gornji tekst bi bio prikazan kao

- ETF
- ostali

Različiti čitači će listu navođenja prikazati drugačije. Za naznačavanje članova liste mogu se koristiti različiti simboli.

- **Liste nabrajanja**

Liste nabrajanja služe da navedemo spisak tema numerisan po redosledu nabrajanja. Sintaksa je identična kao u slučaju *liste navođenja*, osim što umesto oznake <UL> koristimo oznaku <OL>. Ako sada ponovimo prethodni primer, stvari postaju jasnije.

Primer za listu navođenja

```
<OL>
<LI>ETF
<LI>ostali
</OL>
```

Gornji tekst bio bi prikazan kao:

1. ETF
2. ostali

- Opisne liste

Opisne liste počinju oznakom <DL> (Description List) i sastoje se iz naslova (označeno sa DT) i opisa (označenog sa DD).

Primer opisne liste.

```
<DL>
<DT>YUCCA
<DD>YUCCA je Jugoslovensko
udruzenje za racunarske komunikaci-
je. Osnovana je 1994. godine kao
udruzenje gradjana
<DT>LINUX
<DD>PD UNIX, koji je nastao pod
GNU licencom
</DL>
```

Prikazan tekst bio bi formatiran na sledeći način

YUCCA
: YUCCA je Jugoslovensko udruzenje za racunarske komunikacije. Osnovana je 1994. godine kao udruzenje gradjana

LINUX
: PD UNIX, koji je nastao pod GNU licencom

Naslov (<DT>) i opis (<DD>) mogu sadržavati više paragrafa (odvojenih sa <P>) jedan od drugog.

Liste se mogu i ugnježdavati.

Formatiranje reči ili delova dokumenta

Pojedine reči ili delovi teksta mogu se prikazati u različitim stilovima. Stilove uslovno delimo na logičke (način prikazivanja određen je čitačem, primer je <CITE>, <VAR>...) i fizičke (način prikazivanja određujemo sami u tekstu, primer je <I> za prikaz italic, <B> za prikaz **bold**).

Fizički stilovi su:

- TT – fiksirano sa fontom
- B – **bold**, pojačano
- I – italic, iskošeno
- U – podvučeno
- S – tekst se prikazuje sa crtom po sredini
- BIG – tekst se prikazuje većim fontom nego standardno
- SMALL – tekst se prikazuje manjim fontom nego standardno
- SUB – *subscript*, tekst se prikazuje u subskriptu, manjim fontom
- SUP – *superscript*, tekst se prikazuje u superskriptu, manjim fontom

Logički stilovi su:

- EM – isticanje, obično *italic*
- STRONG – jače isticanje, obično **bold**
- CODE – primer koda, obično font sa znacima iste veličine
- SAMP – niz slova,
- KBD – u priručnicima tekst koji treba da otkuca korisnik
- VAR – naziv promenljive
- DFN – definicija, obično **bold** ili ***bold italic***,
- CITE – citat, obično *italic*
- Q – mali citat, obično se interpretira zavisno od jezičkog konteksta, koji se podešava atributom LANG. Za engleski se interpretira kao dvostruki ili jednostruki znak navoda, zavisno od toga da li ima ugnježdavanja citata
- AU – isto kao i AUTHOR.
- PERSON – koristi se za imena ljudi i omogućava da se imena izdvoje automatski od strane indeksirajućih programa.
- ACRONYM – koristi se za isticanje akronima.
- ABBREV – koristi se za isticanje abrevijacija.
- INS – koristi se za ubačen tekst, na primer u prikazu pravnih dokumenata.
- DEL – koristi se za izbačen tekst, na primer u prikazu pravnih dokumenata.

Ako želimo da se pojedine reči prikažu većom ili manjom veličinom fonta to ćemo naznačiti sa

```
<FONT SIZE=x>X je zeljena velicina
fonta!</FONT>
```

A dejstvo promene veličine fonta okončava se znakom </FONT>

A šta kad ne želimo da se tekst formatira?

Već je rečeno da se tekst u HTML dokumentu mora formatirati za ispis, da višak razmaka, praznih redova i drugog nije ne znači i da se ignoriše. Međutim, ponekada nam je veoma značajno da se u ... da se on prikaže onako kako je naveden. Takav tekst ćemo navesti između para oznaka <PRE> na početku i </PRE> na kraju teksta.

Primer neformatiranog teksta.

```
<PRE>
#include <stdio.h>

main()
{
   printf("Zdravo svete!\n");
}
</PRE>
```

On će biti prikazan kako je i naveden, samo bez para oznaka <PRE> na početku i </PRE> na kraju teksta.

Adresiranje

Oznaka <ADDRESS> koristi se u HTML dokumentu da kaže nešto o samom autoru, da najčešće specificira način na koji se sa njim može stupiti u kontakt. Obično se nalazi na kraju dokumenta.

Primer za adresu

```
<ADDRESS>
Uvod u HTML / <A HREF "spar
tic.html">Autor Srdjan Partic</A>
</ADDRESS>
```

Za kraj

Ovo je osnovni skelet HTML-a koji omogućava pisanje prezentacija. Osim toga, za pisanje sopstvenog home page potrebno je samo mnogo mašte i zanimljive slike. ■

Škola jezika C++ (8)

Optimalno, "optimalnije"

U Vujaklijinom leksikonu pod odrednicom "optimalan" stoji "koji najbolje odgovara, najpovoljniji". Latinski koren ove reči ima značenje "najbolji". Dok se na zemlji nisu pojavili programeri, nikome nije padalo na pamet da traži nešto "optimalnije" ili da proglasi nešto drugo za "najoptimalnije". Začudo, ekvivalent ove jezičke greške može se naći i u sorsu programera.

Veliki broj programera živi u zabludi da "zna" kako treba "optimizovati kod pri pisanju", tj. pisati program tako da se izvršava najbrže moguće. Zato se ne retko događa da se unapred odluče za neki komplikovaniji ili nečitljiv način pisanja nečega jednostavnog, u naivnoj nadi da su tako uradili nešto jako dobro, a da kao rezultat dobiju samo ružan kod (problem za kasnije održavanje ili menjanje) koji kao za inat nije ni brži od onog jednostavnije napisanog.

Evo jednog vrlo ilustrativnog primera. Imamo pet funkcija **a**, **b**, **c**, **d** i **e**. Na određenom mestu u programu potrebno je da pozovemo jednu od funkcija, u zavisnosti od parametra **i**: za **i=0**, **1**, **2**, **3**, **4** potrebno je pozvati **b**, **c**, **d**, **a** odnosno **e**. Kako ovo rešiti? Zdravorazumski, jedna **switch** naredba rešava problem. Međutim, to ne može da bude optimalno, rezonuje programer. Da budemo do kraja pošteni, primer smo malo banalizovali: većini programera ovde još ne radi "zvono za uzbunu", ali kada je umesto pet funkcija reč o dvesta, hipotetični programer će se zabrinuti nad ogromnom **switch** naredbom. I trudiće se svim silama da je napravi "optimalnijom" (namerno koristimo neispravnu komparaciju). Ova reakcija je normalno svojstvena nedovoljno školovanom programeru. On je naučio da je konstrukcija

```
switch (i) {
case 2: a(); break;
case 0: b(); break;
case 1: c(); break;
case 3: d(); break;
case 4: e(); break;
}
```

ekvivalentna **if ... else if** konstrukciji

```
if ( i == 2 ) a();
else if ( i == 0 ) b()
else if ( i == 1 ) c()
else if ( i == 3 ) d()
else if ( i == 4 ) e()
```

Programer zaključuje: ako je $i = 4$, onda će se dakle uraditi pet poređenja pre izvršavanja **e()**. A tek sve ovo za dvesta funkcija... I onda smišlja recepte da malo "ubrza posao". Verovatno će prvo smisliti nešto kao "više nivoa, manje poređenja":

```
if ( i >= 2 ) {
switch (i) {
case 3: d(); break;
case 4: e(); break;
} } else {
switch ( i ) {
case 2: a(); break;
case 0: b(); break;
case 1: c(); break;
} }
```

Ako je familijaran sa pokazivačima na funkcije, možda će smisliti i nešto što potpuno eliminiše više poređenja

```
typedef void [illegible]
p_to_f spisak[ [illegible]
void g( int i [illegible]
{
   if ( i ??? 5 [illegible]
     spisak[ i [illegible]
}
```

Gornji kod na prvi pogled izgleda sjajno: deklariše se tip pokazivača na funkcije koje imaju nula argumenata i ne vraćaju rezultat, pažljivo se inicijalizuje niz pokazivača na željene funkcije, a zatim se indeksiranjem izvršava prava. Jedan pristup nizu pokazivača i izvršavanje kroz pokazivač, dakle vrlo efikasno. Skoro bez poređenja: **if** linija je neophodna ako ne želimo da program ispusti dušu za slučaj u kojem **i** nije manje od 5.

Ali u gornjem kodu pretpostavka je bila da sve funkcije imaju isti broj parametara i vraćaju isti rezultat. Ako to nije slučaj, stvar se bitno komplikuje: programer će onda na silu praviti funkcije tako da sve budu identično deklarisane, pa će u svakoj funkciji birati koje će parametre koristiti. Pripremanje parametara biće, međutim, obavezno negde unutar nekih novih **if** konstrukata ili novih **switch** naredbi. Primer rezultujućeg rogobatnog sorsa ovde neće biti prikazan, ali čitalac će ga lako zamisliti. A počeli smo od tako jednostavnog zahteva.

Problematičnost svih modifikacija potiče od pogrešne pretpostavke u izloženom razmišljanju: **switch** jeste logički ekvivalentan sa **if ... else if**, ali to ne znači da će se oba konstrukta izvršavati na isti način!

Konstrukt **switch** svaki C i C++ kompajler kad god može realizuje pri prevođenju na način koji najviše asocira na **ON (I) GOTO ...** konstrukte nekih davnih programskih jezika. "Kad god može" znači da su uslovi navedeni pod **case** takvi da ima smisla praviti tabelu. Evo primera kada nema smisla:

```
switch (i) {
case 30000: a(); break;
case 23001: b(), break;
case 1: c(), break;
}
```

Odgovarajuća tabela skokova imala bi 30000 ulaza, što je očigledno neisplativo. Mora se porediti vrednost po vrednost. Čak, može da se desi da kompajler za ovakav slučaj generiše i nešto neefikasniji kod nego da je gornji uslov napisan pomoću dve **if** naredbe koje ima prilike da sam optimizuje.

Možda izgleda previše smelo tvrdnja o tome da svaki C i C++ kompajler rešava **switch** preko tabela skokova. To je, međutim, istina. Naredba **switch** postoji pre svega da bi programiranje tzv. konačnih automata moglo da bude maksimalno efikasno.

Sada se vratimo na polazne zahteve. Ako funkcije nemaju jednake parametre, rešenje sa eksplicitnim tabelama može da bude neznatno bolje od polaznog **switch** konstrukta, ali u većini slučajeva razlika će biti teško uočljiva, jer se na mašinskom nivou izvršava skoro identičan kod. Zato je **switch** rešenje u prednosti, jer smanjuje šansu uvođenja bagova pri modifikovanju programa: tabelu skokova generiše kompajler, a programer ne mora da je održava i da ostatak koda usklađuje sa njom. Rešenje sa ugnježđavanjem je u svakom slučaju najgore moguće (a nastalo je zato što se programer nadao napamet poboljšanju!). I na kraju, ako funkcije imaju jednake parametre, ubedljivo najefikasnije rešenje je upravo najednostavnije: **switch** rešenje, ono koje je "izgledalo sumnjivo" pa je izazvalo "traženje optimalnijeg". Bez ikakvog našeg upitanja kompajler će interno generisati potrebnu tabelu i najbolje uraditi ono što treba.

Šta je zaključak? Da o optimalnosti sa stanovišta brzine izvršavanja ne treba suditi na osnovu izgleda sorsa, već isključivo na osnovu merenja koja će dati pravu sliku stanja stvari. Ili, da ne treba unapred komplikovati kod na osnovu nekih pretpostavki koje možda i nisu tačne. Kod, za početak, treba da sadrži unutarnji sklad, a posle će se lako ubrzati samo deo za koji se na osnovu merenja ispostavi da je kritičan.

Dafov uređaj

Kritičari jezika C i C++ često baš kao razlog protiv ovog jezika navode čudnu rogobatnost **switch** naredbe: mogućnost "propadanja naniže" kada se pre sledećeg **case** ne stavi **break**. Verovatno najšokantniji primer primene baš ovako zamišljenog konstrukta za pisanje optimalnog programa je tzv. Dafov uređaj.

Smislio ga je Tom Daf, radeći u firmi *Lucasfilm*. Verovatno je svim ljubiteljima *Zvezdanih ratova* poznato da većina specijalnih efekata, u kojima su prefinjene makete pomerane po veoma preciznim putanjama, ne bi bila moguća da svaki pomak kamere ili makete nije bio računarski kontrolisan. Dafu je bilo potrebno da maksimalno ubrza slanje određenih podataka na serijski port. Primetio je da se petlja tipa

```
while ( count-- > 0 )
   *to = *from++
```

može ubrzati ako se izbegne da se posle svakog slanja proverava dokle se stiglo sa brojanjem. I tako Dafov uređaj izgleda ovako:

```
register n = (count + 7) / 8;
/* count > 0 */
switch (count % 8)
case 0:     do {  *to = *from++;
case 7:           *to = *from++;
case 6:           *to = *from++;
case 5:           *to = *from++;
case 4:           *to = *from++;
case 3:           *to = *from++;
case 2:           *to = *from++;
case 1:           *to = *from++;
} while (--n > 0)
```

Ovde se brojač osam puta ređe proverava nego u prethodnoj petlji od dva reda. "Komplikovanje" je u konkretnom slučaju donelo dobitak. U opštem slučaju, da bi bilo opravdano, mora se zaključiti da program pri izvršavanju najviše vremena provede upravo u toj petlji. Gde program koji pišete provodi najviše vremena? S obzirom na moć optimizacije današnjih kompajlera, a još više na kompleksnost današnjih okruženja, ne postoji čovek koji će napamet uvek davati tačan odgovor. Samo pravilno planirani i sprovedeni eksperimenti mogu dati odgovor na takvo pitanje. ■

Janko Stamenović je diplomirani inženjer elektrotehnike na smeru Računarska tehnika i informatika. Trenutno se bavi razvojem složenih aplikacija za Windows u programskom jeziku C++. Možete ga kontaktirati na e-mail adrese janko@sezam.co.yu i janko@galeb.etf.bg.ac.yu

1045 [COMM,WINSOCK;PC]: U skladu sa svetskim trendovima i kod nas je hit sezone među vlasnicima modema komunikacija sa Internetom. Iako još uvek nemamo pristup pravom Internetu i lokalni YUInternet pobuđuje veliku pažnju korisnika. Obzirom na nedostatak pravih providera, razumljiva je velika popularnost programa *Troy Winsock* koji omogućava korišćenje Internet servisa pomoću običnog korisničkog naloga na nekoj Unix mašini vezanoj u Internet. O ovom programu je naš list pisao u više navrata, a pominjan je nekoliko puta i u ovoj rubrici.

Među najčešće korišćenim programima za pristup Internet servisima je i *WS_FTP* za prenos fajlova. Ukoliko koristite *Troy Winsock*, obavezno u dijalogu *Session Profile/Advanced* programa *WS_FTP* uključite opciju *Passive Transfers*.

1046 [OS,WIN95;PC]: Verovatno vam je dosadilo da svaki put kad treba restartovati *Windows 95* čekate da računar obavi celu proceduru. Ukoliko želite da se vaš *Windows* daleko brže restartuje, kliknite na *Start/Shutdown/Restart Your Computer* i onda držeći levi **Shift** pritisnete **Enter**.

[preuzeto sa Sezama, prilog **Nebojša Jevtić** (*jevta*)]

1047 [MREZ,WINDOWS,NOVELL;PC]: Znamo da se CD ROM u *Windows For Worksgroups* mreži deli kroz mrežu tako što se MSCDEX startuje sa opcijom /S. Međutim, ukoliko je mreža bazirana na *Novell Netware* serveru, dešava se da CD ROM ostane nevidljiv na ostatku mreže. Jedan pogled na dokumentaciju po običaju rešava sve probleme - MSCDEX mora da bude učitan PRE mrežnih drajvera IPX (IPXODI) i NETX.

[preuzeto sa Sezama, prilog **Dragan Grbić** (*drgrba*)]

1048 [OTXT;WINWORD;PC]: Kako formatirati brojeve stranica u *Winwordu* 6.0, tj. kako izabrati veličinu i font za njih? Prvo je potrebno pomoću *Insert/Page Numbers* odabrati mesto i početni broj za numeraciju. Zatim izaberite opciju *View/Header And Footer*. Pomoću prve ikonice izaberite *header* ili *footer* zavisno od mesta koje ste odredili za brojeve strana i mišem selektujte sam broj. Pojaviće se tekući format (vrsta i veličina fonta) koji se može menjati. Promene koje napravite imaju efekat i za sve ostale strane.

[preuzeto sa Sezama, prilog **Slobodan Perić** (*peri*)]

1049 [COMM,FAX;PC]: Jedan od osnovnih prigovora koje domaći korisnici upućuju raznim *fax* programima je nemogućnost slanja *fax*-a bez prethodnog biranja telefonskog broja. Izgleda da autori tih programa nisu mnogo vodili računa o siromašnijim zemljama u kojima nije jednostavno odvojiti posebnu telefonsku liniju samo za *fax*. Zbog toga se često koriste programi poput *RipeFax* ili *BitFax for Windows* koji imaju ovu mogućnost, ali je manje poznato da se i većina drugih programa može prevariti na ovaj način. Dovoljno je da umesto telefonskog broja unesete zarez - program smatra zarez za deo broja, pa ne prijavljuje grešku.

[preuzeto sa Sezama, prilog **Vladan Marinković** (*vlad*)]

1050 [OS,MSDOS;PC]: Iako mu *OS/2*, *Windows* i razne verzije *Unix*-a odavno prete, za DOS još uvek vlada prilično interesovanje, pa možda nije loše prisetiti se jednog malog trika vezanog za BATCH programe. Ovi programi se često koriste za instalaciju softverskih paketa, pa je često potrebno proveriti postojanje direktorijuma. Standardan metod IF EXISTS radi samo u slučaju fajlova, ali ne i direktorijuma, pa se ne može primeniti u ovu svrhu. Sa druge strane, osobina DOS-a je da svaki direktorijum sadrži i "nevidljivi" fajl *nul*. Prema tome, ukoliko želite da iz BATCH programa testirate postojanje direktorijuma DIR, upotrebićete naredbu IF EXISTS C:\DIR\nul.

1051 [OS,WINDOWS;PC]: Da li igrate *Hearts* koji se isporučuje uz mrežne verzije *Windows*-a? Igra je stekla priličnu popularnost zato što se može igrati i u mreži. Moguće je i varanje, ukoliko u odgovarajuću sekciju WIN.INI upišete redove

```
zb=42
gamemeister=1
```

, zatim u toku igre pritisnete **Shift-Ctrl-Alt-F12**, moći ćete da vidite karte ostalih igrača.

Moguće je i definisati imena igrača koje inače dodeljuje sam kompjuter (kad se ne igra u mreži). Dovoljno je u istoj sekciji navesti imena pomoću redova

```
p1name=
p2name=
p3name=
```

[preuzeto sa Sezama, prilog **Nenad Čanić** (*cnenad*) i **Milan Dražić** (*mdrazic*)]

1052 [OTXT,WINWORD;PC]: Ukoliko ste nedavno prešli sa neke od verzija *Word Perfect*-a na *Word*, verovatno vam nedostaje opcija *Flush Right* koja deo linije desno od kurzora uravnava po desnoj margini. *Word* nema ekvivalent te opcije, već uravnavanje izvodi pomoću tabulatora: odaberite *Format/Tabs* i tamo definišite jedan tabulator uz desno poravnanje (*Alignment Right*). Možete definisati i stil sa ovakvim tabulatorima i koristiti ga kad vam je *Flush Right* potreban. Uz to, možete odrediti i *leader* karakter kojim će biti popunjen prostor između tabulatora i dela linije poravnatog po desnoj margini.

1053 [OS,OS2;PC]: Ukoliko želite da koristite *Windows* aplikacije u *seamless* modu, dakle u prozoru na *OS/2* desktopu, potrebno je da drajver za video kartu to podržava. Svi drajveri koji se dobijaju uz *OS/2* podržavaju *seamless* mod i najbolje je koristiti njih.

Ukoliko nemate *OS/2* drajvere za vašu video kartu, možete koristiti i *Windows* drajvere, ali će *Windows* programi tada raditi samo u *full-screen* modu. Ipak, uz većinu kartica koje se mogu kupiti kod nas dobijaju se i *OS/2* drajveri.

[preuzeto sa Sezama, prilog **Nenad Milenković** (*wizard*)]

1054 [OS,OS2;PC]: Ukoliko vam se dešava da neki DOS ili *Windows* program ostane u memoriji, pa ne možete regularno da izađete iz *Warp*-a, resetovaćete mašinu, ali da vam se ne bi desilo da *Warp* sam ponovo startuje nekog od njih, dodajte u *Warp*-ov CONFIG.SYS red

```
set restarto-
bjects=startupfoldersonly
```

[preuzeto sa Sezama, prilog **Novica Milić** (*novim*)]

1055 [MREZ,WINDOWS;PC]: Kako povezati stanicu bez lokalnog diska u *Windows for Workgroups* mrežu? Microsoft prodaje poseban paket za DOS koji se zove *Workgroup Add-On for MS DOS*. Isto se može izvesti i uz pomoć standardnog *Windows for Workgroups* paketa, naravno vodeći računa o licenci za višestruko korišćenje iste kopije.

Potrebno je da obezbedite da bar jedna od *Windows* stanica ima iste parametre mrežne kartice kao i *diskless* stanica. Zatim na toj *Windows* stanici bekapujete fajlove AUTOEXEC.BAT, CONFIG.SYS, SYSTEM.INI i PROTOCOL.INI. Onda startuje-

Windows u njemu Network Setup. Kao protokol postavite *NetBEUI*, a za drajver za mrežnu karticu definišete *Real Mode NDIS Driver*. Zatim snimite konfiguraciju, izađete u DOS i na disketu sa koje će se diskless stanica podizati u poseban direktorijum (recimo A:\NET) iskopirate sledeće fajlove: NET.EXE, NET.MSG, NETH.MSG, PROTMAN.DOS, PROTMAN.EXE, SYSTEM.INI, PROTOCOL.INI, IFSHLP.SYS, NDISHLP.SYS i sam drajver za mrežni adapter, koji obavezno ima ekstenziju .DOS (za klonove NE2000 kartice, ime fajla je NE2000.DOS). Nakon toga ne zaboravite da vratite bekapovane konfiguracione fajlove na *Windows* stanici.

Pre startovanja diskless stanice potrebno je još da editujete fajl SYSTEM.INI. U pitanju je standardni *Windows* konfiguracioni fajl, ali možete obrisati sve sekcije osim [*Network*], [*Network drivers*] i eventualno [*Passwords*]. U sekciji [*Network*] obavezno promenite ime korisnika i kompjutera, inače ćete pri prijavljivanju na mrežu dobiti poruku o grešci. U sekciji [*Network drivers*] podesite opciju **devdir** tako da odgovara putanji na disketi. Potrebno je još dodati u CONFIG.SYS liniju **device=A:\NET\IFSHLP.SYS**, i u AUTOEXEC.BAT **NET START WORKSTATION**.

Ovako konfigurisana stanica može bez problema da radi kao klijent i da pristupa svim diskovima i štampačima u mreži. Isti princip može se primeniti i za mašine koje imaju hard-disk, ali ne mogu da izvršavaju *Windows*, npr. 286 mašine.

1056 [OTXT,WINWORD;PC]: Još nekoliko zanimljivosti vezanih za *Winword*. Nova verzija (7.0) ovog programa ima jednu zabavnu osobinu: ukoliko ste navikli na "jezik" BBS-ova, pa otkucate smajli :), *Word* će ga zameniti simbolom "*happy face*" (srećno lice) u *Times New Roman* fontu. Odgovor je, naravno, u *AutoCorrect*-u - tamo je definisana transformacija smajlija u "*happy face*", kao i mnoštvo lepih caka.

[Preuzeto sa Sezama, prilog **Čedomil Višnjić** (*fancy*)]

U okviru istog "bajta" dodaćemo još jednu zanimljivost vezanu za *Word*. Do sada je bilo poznato da je *Word* jedan od najkvalitetnijih, najpopularnijih, ali i najzahtevnijih tekst procesora za *Windows*. Međutim, nedavno smo otkrili i da je jedan od najnestrpljivijih.

Sa dolaskom *Windows*-a 95 pojavilo se mnoštvo programa koji "starim" 16-bitnim programima omogućavaju korišćenje dugih imena fajlova. Međutim, neki izazivaju vrlo neobično ponašanje *Word*-a 6.0. Recimo, probali smo jedan od takvih programa, *WideOpen*, koji se instalira kao macro koji zamenjuje standardne *Word* dijaloge za rad sa fajlovima novim *Windows 95* dijalozima. Sticajem okolnosti desilo se da je dijalog za otvaranje fajlova stajao na ekranu 3-4 minuta, a onda je *Word* počeo čudno da se ponaša: *title-bar* je počeo da flešuje. Nakon konačnog unošenja imena fajla *Word* je ispisao poruku: *This task is taking longer than expected. Do you want to continue waiting?* Ovo je delovalo krajnje zbunjujuće, a pogled u README fajl programa *WideOpen* otkrio je da je i autor tog programa veoma zbunjen ovakvim ponašanjem *Word*-a. Čak je i postavio pitanje: zna li neko način da se *Word* malo nauči strpljenju?

1057 [OS,WIN95;PC]: Ako želite da jedan ili više označenih fajlova obrišete i pri tom izbegnete njihovo premeštanje u *Recycle Bin*, pritisnite desni taster miša držeći **Shift** i izaberite *Delete* u meniju koji se pojavi. Interesantno je, inače, da kombinacija tastera SHFT + DEL ne funkcioniše.

[Preuzeto sa Sezama, prilog **Aleksandar Petrović** (*iznogud*)]

1058 [OS,WIN95;PC]: Ako pokušate da pročitate neki *help*, javiće vam se poruka tipa *Preparing Help file for first use* ako ga od instaliranja *Windows*-a 95 niste pokretali. *Windows* će napraviti fajl istog imena kao i .HLP fajl, samo sa ekstenzijom .GID. Veličina .GID datoteke zavisi od veličine .HLP fajla, za velike helpove može dosta da naraste (recimo, za *bwinapi.hlp*, koji je dugačak oko 9MB, *bwinapi.gid* je dugačak oko 450 Kb). Lepa stvar je što .GID fajl možete slobodno da obrišete ako ne nameravate da upotrebljavate dati *help* sledećih dana - .GID fajl će se sam generisati kada budete sledeći put otvorili help, uz odgovarajuću poruku o "pokretanju helpa prvi put". Najzanimljivija stvar sa .GID fajlovima je ta što su skriveni, tj. imaju setovan *hidden* atribut, tako da ih je teško zapaziti.

[Preuzeto sa Sezama, prilog **Aleksandar Petrović** (*iznogud*)]

1059 [OS,WIN95,PC]: Iako bi svi programi sa *Windows* 3.11 trebalo da rade i sa *Windows* 95, neki od njih prave probleme. Evo spiska problematičnih programa sa opisom grešaka. ■

Program	Proizvođač	Problem
Illustrator 4.0	*Adobe*	Ako printer koristi EMF [illegible]
Photoshop 3.0	*Adobe*	Promena vrednosti u prozorima [illegible]
C++ 4.0 4.02	*Borland*	Primeri OLE programa mogu da [illegible]
cc:Mail for Windows	*Lotus*	U nekim slučajevima neispravno očitava [illegible]
CorelDraw 4.0	*Corel*	Kod izvoženja u .BMP, .PCX ili TIFF format [illegible]
Corel Ventura 4.2	*Corel*	Pri skrolovanju na ekranu se pojavljuju horizontalne linije
dBase 5.0 for Windows	*Borland*	*Windows* 95 Help prijavljuje višestruke probleme
Hijaak PRO 2.0	*Inset Systems*	Nije u stanju da "zarobi" ispis DOS-a na štampač
LapLink for Windows 6.0	*Traveling Software*	Ne radi pod *Windows* 95
Director 4.04	*Macromedia*	Prozori sa paletama se ponekad pogrešno regenerišu
Norton AntiVirus 2.0, 2.1	*Symantec*	Ne radi pod *Windows* 95
Norton Backup 3.0 for Windows	*Symantec*	Ne radi pod *Windows* 95
Norton Desktop for Windows	*Symantec*	Ne radi pod *Windows* 95
Norton Utilities 8.0 for Windows	*Symantec*	Ne radi pod *Windows* 95
pcAnywhere	*Symantec*	Zahteva MS DOS mod da bi radio
PC Tools	*Symantec*	Ne radi pod *Windows* 95
PerfectOffice 3.0	*Novell*	Pokretanje ma kog programa ispisuje prompt za logovanje na mrežu
Sidekick 2.0 for Windows	*Starfish Software*	Javljaju se problemi sa prikazom slike na monitoru
Stacker	*Stac Electronics*	Ne radi pod *Windows* 95

Igra života

"Igra života" se odigrava na tabli veličine 8*8 u kojoj se u svakom polju može nalaziti po jedna ćelija. Pravila navedena u pitalici RI11 su definisala oživljavanje, odnosno umiranje ćelije u sledećoj generaciji u zavisnosti od broja susednih živih ćelija: mrtva ćelija oživljava ako ima tačno tri živa suseda, dok živa umire ako ima manje od dva ili više od tri živa suseda. Pod susednim ćelijama računaju se i dijagonalna polja tabele. Dodatni uslov je bio da živa ćelija obavezno umire u sledećoj generaciji ako se nalazi na nekom od ivičnih polja. Trebalo je sastaviti takav raspored živih i mrtvih ćelija da se u sledećoj iteraciji dobije što više živih ćelija na tabli.

Rešavači su odmah odustali od primene "grube sile". S obzirom da ukupan broj kombinacija (različitih rasporeda ćelija na tabli) iznosi 2^64, jasno je da se pre sedanja za računar mora izvršiti racionalizacija problema. Iako su rešavači do rezultata dolazili različitim rezonovanjima, većina od njih je dospela do istovetnog krajnjeg oblika, koji je prikazan levo na slici.

Najpre treba imati u vidu da rešenje (ono iz kojeg se u sledećem koraku dobija najveći broj živih ćelija) treba da ima što manje živih ćelija na ivičnim poljima, s obzirom da iste umiru u sledećoj generaciji. Pošto će ivična polja biti prazna, najzgodnije je da se žive ćelije postave u njima susedne redove. Ako ceo drugi red ispunimo živim ćelijama, a prvi i treći ostavimo praznima, ćelije iz drugog reda će ostati žive (zato što svaka ima levog i desnog suseda) dok će one iz susednih redova oživeti jer se graniče sa po tri ćelije iz drugog reda. Do problema dolazi jedino na krajnjim ćelijama redova, što se rešava malo drugačijom organizacijom ovih polja - pogledajte sliku rešenja. S obzirom na veliku efikasnost ovakve postavke, možemo sa opravdanom sigurnošću smatrati da je ona i najbolja moguća. Ako istu logiku primenimo na sedmi red, kao i na drugu i sedmu kolonu, jedino što nam ostaje je neispitan kvadrat veličine 2*2 u samom središtu table, za koji je lako proveriti da ga je najbolje ostaviti praznog.

Ovakvim postupkom dolazimo do optimalnog rasporeda koji u sledećoj generaciji proizvodi tablu sa 52 žive ćelije, prikazanu na desnoj strani slike. Strogo posmatrano, da bismo bili sigurni da je ovo rešenje zaista najbolje, trebalo bi ispitati sve preostale kombinacije ili naći drugi precizan način da se to potvrdi. Ovako daleko, jasno, ne treba uvek ići, iako bi nam neki matematičar možda zamerio. Za potrebe Pitalica dovoljno je da nam je intuitivno jasno da se radi o najboljem rešenju.

Nagrade

Najbolje obrazloženo rešenje poslao je **Vinko Marinković** iz Beograda, koji je osvojio prvu nagradu u iznosu od 100 dinara i osvojio 3 boda u godišnjem takmičenju. G. Marinković je bio i najbolji rešavač pitalice RI10, tako da sa ukupno 6 bodova ubedljivo vodi na našem godišnjem takmičenju rešavača.

Miloš Prvulović iz Kumodraža je zaslužio drugu nagradu u iznosu od 50 dinara i 2 boda u godišnjem takmičenju, dok je među preostalim uspešnim rešavačima najviše sreće imao **Zoran Bojić** iz Kraljeva i dobio treću nagradu koja mu donosi 25 dinara. Njemu, kao i svima koji su poslali tačno rešenje, pripada 1 bod u godišnjem takmičenju.

Na kraju, jedna lepa vest: nagrade će, počevši od ove pitalice, biti uvećane, tako da će rešavači dobijati 200 dinara za prvo mesto, 100 za drugo i 50 za treće. Nadamo se da će nove nagrade, kao i novi izazovni zadaci, privući još više čitalaca ovoj rubrici. ■



Rešenje pitalice



Sledeca generacija

Zadatak RI13

Najbliža reč

Najpre ćemo svim slovima azbuke dodeliti odgovarajuće brojne vrednosti po azbučnom redu: slovo A dobija vrednost 1, slovo B 2, V 3, G 4 i tako dalje do slova Š koje ima vrednost 30. Zatim definišemo "brojnu vrednost" svake reči kao proizvod svih vrednosti slova: PERA = 19*7*20*1 = 2 660. Potrebno je naći reč koja ima vrednost što bližu reči RAČUNARI. Da bi reč bila priznata kao rešenje, potrebno je da postoji u srpskom jeziku i da ne bude izvedenica glagola "računati". Neću vam postavljati previše stroge uslove niti navoditi nekakve referentne rečnike - dozvoljene su sve vrste reči u svim padežima i oblicima, u ekavskom ili ijekavskom narečju, tuđice... samo obratite pažnju da ne pravite neuobičajene kovanice ili izmišljate neke [illegible] nazive.

RI13: Najpribližnu vrednost ima reč ______

Ime i prezime ______

Adresa ______

Mesto ______

Računar ______

Vreme ______

Identifikacioni broj ___ ___ ___ z broja telefona ___ ___ godina rođenja ___ ___ po izboru

Kako poslati rešenje

Rešenja pitalice, zajedno sa obrazloženim [illegible] adresu *Računari* (za *Pitalice*) Bulevar vojvode [illegible] najkasnije **pre 20. decembra 1995. godine**. Sva [illegible] pravo na novčane nagrade od 200, 100, 50 dinara, [illegible] (kupone) na kojem je upisan identifikacioni broj [illegible] rešavača Pitalica. Obratite pažnju na to da sva rešenja [illegible] identifikacioni broj. Osim običnom [illegible] rešenja [illegible] elektronskom poštom posredstvom Sezama [illegible] komande [illegible] REŠENJE [illegible] iz komandnog moda pošaljite ličnu poruku [illegible]. Zadatak i uputstvo za slanje rešenja su kuponom [illegible] PITALICE[illegible].txt. [illegible]

SEZAM

SISTEM ZA MODEMSKE KOMUNIKACIJE

Pripremio: Dejan Ristanović, uz pomoć moderatora konferencija

Moderatori: Vladan Aleksić, Dejan Balinda (CIVILIZACIJA), Nikola Bošković (PC USER), Jovan Bulajić (FILES), Dragan Čukanović (JURE), Vlada Ćalić (MUZIKA), Dejan Jelović (PC PROG), Darko Jevtović, Saša Krstić (PC HARD), Slobodan Kalezić (ORKA), Bojan Pavković (PC UTIL), Pavle Peković (PC.OS), Miloš Prvulović (AMIGA), Dejan Ristanović (NOVOSTI, KOMUNIKACIJE), Darko Stanišić i Željko Krstić (MCEVI), Bojan Stanojević (ATARI), Bojan Tepavčević (FORUM), Zoran Životić (SEZAM).

Sezam je u oktobru opslužio 37.652 poziva, komunicirao sa korisnicima 7161 čas, odnosno 298 dana (u proseku je 9,6 nodova bilo **neprekidno** zauzeto), od čega je u chat-u provedeno 626 sati, ili 78 radnih dana. Korisnici su sa Sezama preuzeli 1,76 gigabajta podataka, dok su za isto vreme sistemu poslali skoro 33 megabajta.

Poruka uvek ima puno - tokom oktobra stiglo je 10750 poruka, odnosno 9,36 megabajta teksta, a tome treba dodati i 12781 privatnu poruku (11,6 megabajta) i 4724 poruke u grupama (2,3 megabajta teksta, plus brojne datoteke priključene uz poruke). Najviše poziva uputili su Lena Petković (*queen*), Đorđe Karan (*kum djole*) i Boško Koprivica (*bosko*), najviše vremena na vezi su bili Ivan Jekić (*saboteur*), Lena Petković i Aleksandar Mirčić (*mirche*), u chat-u su (uz njih) najduže boravili Dejan Krstić (*dekk*), Vera Mijanović (*mvera*), fajlove su najviše prenosili Goran Gavrilović (*gorang*), Andor Ginder (*pishta*) i Miodrag Marković (*micix*), dok su najviše konferencijskih poruka poslali Stanislav Žunjić (*supers*), Brana Tomić (*bane*) i Čedomil Višnjić (*fancy*) - rezultat žestoke diskusije u konferenciji FORUM.

SEZAM

Kako se bližio Sezamov šesti rođendan (oficijelno 11. novembar, premda je Sezam krenuo u probni rad 25. oktobra 1989. godine), tako je i poruka u konferenciji SEZAM bilo sve više. Ovo je za mnoge korisnike bila dobra prilika da prokomentarišu budućnost Sezama i ukažu na razne nedostatke njegove organizacije i održavanja. Razgovaralo se o osnovnoj koncepciji sistema, hardverskom okruženju, problemima koje ono izaziva brzim modemima, Sezam test sistemu, njegovom povezivanju, visini pretplate i mnogim drugim pitanjima.

KOMUNIKACIJE

U konferenciji ponovo stiže puno poruka, najviše ih je pristiglo u temu **modemi**, u kojoj je raspravljano o veoma širokom spektru problema. U žiži su i dalje Voice modemi, koji mnoge privlače ali koje nije uvek lako pokrenuti. U "susednoj" temi **fax** bilo je dosta reči o razlikama između klase 1 i 2, posebno sa aspekta programa *PipeFax*. Predstavljen je program *3D Fax* firme *Infolmaging* i tražena mogućnost slanja faksa iz komandne linije.

Od tema koje se odnose na komunikacione programe najživlje je, kao i obično, bilo u temi **telemate** - neki od problema koje smo rešavali bi verovatno izazvali nevericu na nekom stranom sistemu, ali šta da se radi kada je naokolo toliko centrala na kojima treba čekati *dial tone* po par minuta... Od srodnih tema posećena je bila jedino **kom.programi** gde smo naučili razne stvari o programu *WinComm PRO* firme *Delrina*.

Posle dužeg vremena oživela je tema **hackers** - opisano je nekoliko provala na domaće sisteme (ne samo ona legendarna na BBS Kučevo) uz povremeno neraspoloženje prema sistem administratorima koji su na te provale reagovali.

U temi **email** se, pred kraj meseca, konačno smirila maratonska rasprava koju smo prenosili sa SetNET-a odnosno YU Interneta. Učestvovao je naravno u njoj i poneki Sezamovac, ali je glavnina poruka bilo "prepucavanje" SysOp-a nekoliko BBS-ova o raznim stvarima koje su se dešavale i koje se dešavaju na našoj sceni. U ostalim temama je tražena terminal emulacija za pristup RCUB-u i pristup *public* direktorijumu UBBG-a, rešavan problem slanja faksa sa nekih centrala u inostranstvo (impulsi tako jako otkucavaju da modemi ne mogu da uspostave vezu), opisan pristup *Web*-u iz *Windows 95*.

PC HARD

U oktobru je najaktuelnija tema bila **diskovi**, gde se najviše diskutovalo o bagu u kontroleru hard diska na pojedinim PCI pločama. Zaključeno je, a i sam *Intel* je objavio takvu vest, da korisnici Windows-a 95 nemaju nikakvih problema. *Windows 3.11* je bezbedan samo ako je drajver za 32-bitni pristup proizvod neke od renomiranih firmi kao što su *Ontrack*, *WD*, *EZ Micro house* i sl. Napomenimo da se defektni čipovi ne nalaze na svim PCI pločama, a njihovo prisustvo se najlakše detektuje programom *CTRLTEST* koji je poslao jedan od korisnika.

U temi **sistemi** korisnici su iznosili prve utiske o tek pristiglim Pentium pločama sa *Triton* čipsetom kao i razlikama između nekoliko *Intel* ovih ploča baziranih na ovom čipsetu. U temi **video** se raspravljalo o novim čipovima firmi *S3* i *Cirrus Logic* (5440 i 768/868) sa MPEG podrškom. Tema **kvarovi** je, kao i uvek, bila puna problema nastalih zbog nepodešenosti parametara BIOS-a, što naravno nisu pravi kvarovi, a rešavani su uglavnom savetima iskusnijih hardveraša. U temi **nabavka** su se uglavnom iznosila pozitivna/negativna iskustva prilikom kupovine u našim firmama, pre svega oko kvaliteta proizvoda, stručnosti kadrova i garancije, a neki korisnici su predložili pravljenje crne liste dilera.

PC.OS

Oktobar mesec je, sasvim očekivano, obeležio *Windows 95*. Diskutovalo se o svim aspektima novog operativnog sistema, počev od instalacije do konkretnih problema u radu, o aplikacijama kao i mogućnostima i slabostima *Win 95*. Ova priča je zauzela čak četiri teme u konferenciji PC.OS. Pored novootvorene win.95 i o DOS delu (MS DOS 7.0) novog Windows-a pričalo se u temi **ms.dos**, poređenje sa drugim operativnim sistemima je našlo mesto u temi **os.wars**, dok je priličan broj poruka završio i u temi **windows** koja je ostavljena za diskusije brojne korisnike prethodnih verzija *Windows*-a.

U temi **os.wars** se približavanjem kraja meseca rasplamsala diskusija za koju očekujemo da će se nastaviti bar istom žestinom i u novembru. Teme **os.2** i **unix** su bile uobičajeno žive, i u njima su se ponavljale priče o večnim problemima particionisanja diska, instalacije i konfigurisanja *OS/2*, *Linux*-a tj. *SCO Unix*-a. Za *OS/2* je korisnik novim po običaju poslao velik broj korisnih PD i SW programa.

PC.PROG

Oktobar je u konferenciji PC.PROG protekao mirno, bez ikakvih velikih rasprava i svađa, ali sa puno korisnih informacija. U temi **algoritmi** bilo je priče o matematičkim rutinama za C i C++ kao i o algoritmima za brzu pretragu teksta, dok se u temi **assembler** pričalo prvenstveno o *DMA* prenosu podataka.

Pascal programeri bavili su se algoritmima za crtanje lukova, pristupom komunikacionim portovima, slanjem YU fontova na štampače, *DPMI* standardom, čitanjem serijskog broja diska iz DOS prozora pod Windows-ima, programiranjem *Sound Blaster*-a i direktnim brljanjem po disku u *Delphi*-ju.

C i C++ programeri bavili su se korisničkim interfejsom, dobrim knjigama za učenje jezika, dobrim dizajnom programa, ali je ubedljivo najviše pažnje odnelo traženje dobrog načina za učitavanje podataka iz datoteke, jer se scanf tu pokazao tragično lošim kada treba učitavati brojeve i stringove u isto vreme.

U temi **baze podataka** je, u Americi veoma popularni *PowerBuilder* najzad dobio malo zaslužene pažnje, dok su se Kliperaši uglavnom bavili pretraživanjem teksta, novom verzijom 5.3, višejezičkim programima i problemima sa verzijom *Netware Lite*.

PC.USER

TSE editor je bio zastupljen u temi **editori** sa par fajlova koji su sadržavali razne korisne makroe vezane za sam program. Tekstovi pisani u *Chi Writer*-u 3.0 se ne mogu prebaciti u noviju verziju istog programa 4.0. Ali, rešenje je stiglo i za taj program. Korisnici su našli program koji vrši konverziju iz starog tekst formata u novi.

Kako to biva, tema **word** je bila najposećenija. Rešavali su se stari, neki stari i novi problemi vezani sa radom u samom programu. Insertovanje slika iz *Corel*-a, teksta iz *Corel Venture*, sve smo rešili na zadovoljstvo korisnika koji su imali probleme.

U *Excel*-u smo rešavali prikaz vrednosti koje ispunjavaju određene uslove različitim bojama.

Pojava nove demo verzije popularnog programa *3DS* je malo uzburkala strasti u temi **cad**. Međutim, ograničenje ove verzije programa, koja je još uvek u beta verziji, je to što će raditi samo na mašinama na kojima je instaliran *Win NT*, i to u verziji najmanje 3.5. Dobili smo i tekst vezan za softver *PRIMAVERA*, kao i *VIBRANT* drajvere za *3DS 3/4* od juna 95.

Tema **razno** je donela program domaće proizvodnje *LAT* u verziji 2.0. Program služi za učenje latinskog jezika. Ubrzo posle ove verzije je stigla verzija 2.1 koja je sadržavala neke sitnije ispravke vezane za program, a na koje su pažnju skrenuli sami korisnici Sezama koji su isprobavali program.

FORUM

Uz eho gubitka gradova u Zapadnoj Bosni, FORUM je ušao u mesec oktobar. Pored Sanskog Mosta, Krupe, Ključa, Mrkonjića i ostalih gradova, pažnju korisnika tokom meseca zaokupljao je atentat na Kiru Gligorova, kao i perspektiva mirovnih pregovora u Dejtonu. Međutim, daleko najviše prostora zauzela je rasprava sa jednim autentičnim predstavnikom srpskih neofašista. Postoji velika opasnost da se rasprava prenese i u mesec novembar.

CIVILIZACIJA

Poput prethodnih godina, jesen je u konferenciju CIVILIZACIJA dovela nove učesnike. Uz povratnike sa godišnjih odmora i školskih raspusta, njihovo učešće značajno je doprinelo raznovrsnosti razgovora kakvoj smo se svi uželeli tokom letnjih meseci.

Kako upotreba jezika utiče na vrednosni sadržaj i kakve se zloupotrebe ovde mogu očekivati, bila je dilema kojoj je više učesnika u diskusiji ponudilo primere iz svakodnevnog života. Tema **religija** pratila je obeležavanje značajnih pravoslavnih praznika tokom ovog perioda, a pažnju je privukla i poruka u kojoj je dat osvrt na opšta pravila ponašanja u crkvi. U temi **istorija** obeleženo je nekoliko značajnih datuma iz naše istorije XX veka. **Nauka** je najviše pažnje posvetila načinu korišćenja separatora pri pisanju brojeva.

Ovogodišnji beogradski Sajam knjiga propratila je tuga zbog smrti velikog pisca Slobodana Selenića. U temi **književnost**, pored već rečenog, upoznali smo se i sa ovogodišnjim dobitnikom Nobelove nagrade za književnost, irskim pesnikom Šejmasom Hinijem (*Seamus Heaney*). Pročitali smo šta Bogdan Tirnanić misli o BITEF-u, a zapažena je i izložba ekslibrisa. Bilo je reči i o filmskom repertoaru beogradskih bioskopa tokom proteklog leta. Jezičke rasprave ponovo su bile prepune polemičkih tonova, a tema **lični stav** posle višemesečne umerene aktivnosti prosto je "eksplodirala" u polemici u vezi sa izborom kućnih ljubimaca. Ljubitelji pasa favorizovali su neposrednost koju krasi ova prijatna životinja, dok su pristalice mačaka insistirale na dubljoj individualnosti svojih izabranika. Mada je u oba "tabora" povremeno vladala zbrka između opšteg i posebnog, ova rasprava je bila veoma prijatna s obzirom da je u njoj učestvovalo i prilično "posmatrača" koji su čitav događaj shvatili kao odličnu priliku za dobru šalu. Možda je najbliže istini reći da je i prisustvo u konferenciji CIVILIZACIJA tokom ovog perioda, bilo ponajpre - prijatno!

MUZIKA

U oktobru je konferencija MUZIKA proslavila svoj prvi rođendan i tom prilikom je obogaćena sa preko 800 novih poruka, za šta su uglavnom zaslužne vrlo zanimljive rasprave u temi **rock**. Tokom [illegible] ve [illegible] ugačka diskusija o [illegible] grupe *Queen*.

Teme **klasika**, **jazz**, **blues** i **pop** bile su mirne, a u temu **techno** je zalutala zanimljiva rasprava o uticaju reggae zvuka (i ideologije) na punk i njegove derivate, koja je kasnije preseljena u odgovarajuću temu. I tema **hard heavy** je bila bogata raznim diskusijama od kojih je najupečatljivija bila ona koja se bavila u ovim okvirima nezaobilaznom temom - grupom *Iron Maiden*, uz nekoliko svedočanstava o njihovom koncertu održanom sredinom meseca u Budimpešti. Tema **punk** je bila mesto gde se prenela rasprava o uticaju reggae-a na punk, naročito na grupu *Clash*. Ovo je sprovociralo i razgovore na temu raznih zvaničnih i poluzvaničnih izdanja benda koje se ne ubrajaju u regularne albume, kao i pojavu određenog broja tekstova njihovih najpoznatijih pesama.

Tema **alter** je zašla u mirniju fazu i možda prvi put od otvaranja konferencije nije bila prva po broju pristiglih poruka. Najzapaženije poruke su se odnosile na novi album *Sonic Youth*, priču o ljubljanskoj grupi *Mladi Youneed*, uz tekstove *Meatpuppets* a, *Wedding Present*, *Bauhaus* i drugih.

Rasprave o domaćoj muzici su primetno živnule, pa i tema **yu zvuk** više ne tavori prazna kako je to bilo običaj u ranim danima njenog postojanja. Bilo je dosta komentara novih i starih izdanja raznih domaćih grupa, a najviše se pričalo o grupi Smak. Bilo je i neuspešnih pokušaja oživljavanja odavno mrtve teme **subkultura**.

Tema **razno** je potpuno opravdala svoj naziv, a posebnu pažnju, kao i razne komentare, izazvao je početak redovnog objavljivanja jedne od top-lista Radio Index-a ("Idi mi, dođi mi top-lista" subotom od 9-14, Radio Index 88.9MHz) uz mogućnost glasanja putem Sezamove privatne pošte. Bilo je odobravanja ali i protesta, ali čini se da je ova praksa ipak zaživela, nadamo se na obostranu korist. U svakom slučaju veliki broj poruka u temi razno odnosio se upravo na komentare same liste.

VICEVI

U septembru smo glasali za najbolje viceve u protekla tri meseca, i izabrali četiri pobednika. Junior pobednik: zašto Mujo ima perje oko usta? Šmrče koku. Julski laureat: Stoji crnac u metrou i čita novine na hebrejskom. Posmatra to jedan belac duže vreme i na kraju priđe crncu, uhvati ga za ruku i kaže: "Je l' bre, jel' tebi nije dovoljno što si CRNJA!"

Proverili smo znanje pesama narodnjaka, poslušali dosta emisija "Želje i pozdravi" i dobili source *Microsoft* ove marketinške kampanje (isečak):

```
        if    (latest_window_ver-
sion>one_month_old)
  {
    if (there_are_still_bugs)
      market(bugfix)
    if (sales_drop_below_certain_po-
int)
        raise(RUMOURS_ABOUT_A_NEW_BU-
GLASS_VERSION)
  }
```

I kada smo već kod vrelog *Win95*, znate li koji se taster najviše koristi u njemu? Reset taster. ■

KAKO POSTATI ČLAN

Postupak učlanjenja počinje zvanjem Sezama - podesite parametre komunikacije na **2400 8N1** (ako imate MNP modem, uključite MNP), pozovite (011) 648-422. Posle pozdravne poruke Sezam će vas pitati:

Username:

Na vama je samo da otkucate NEW (novi korisnik) a Sezam će vas dalje voditi kroz proces prijavljivanja.

Izbor imena

Jedan od prvih podataka koje treba da date je puno ime i prezime, a potom i pseudonim pod kojim ćete koristiti usluge Sezama. Jednom izabrani pseudonim ostaje vaše trajno ime na Sezamu (ne može se menjati), što znači da ga treba pažljivo odabrati; predlažemo da se još pre poziva opredelite za pseudonim koji bi trebalo da podseća na vaše ime i prezime. Ukoliko nemate ideja, Sezam će vam predložiti pseudonim sastavljen od vašeg prezimena i prvog slova imena. Pseudonim može da ima između 4 i 16 znakova (preporučuje se 5 do 8 slova, pošto više znakova znači više kucanja, kako za vas, tako i za one koji sa vama komuniciraju); u njemu se mogu nalaziti isključivo slova i znak tačka.

Izbor lozinke

Sledeći korak je izbor lozinke (password) kojom štitite vaš račun. Za razliku od pseudonima koji je javna informacija, lozinku znate samo vi i Sezam. Zato je odaberite pažljivo: naročito izbegavajte da lozinka bude jednaka pseudonimu, ili da se sastoji od vašeg imena, rođendana, imena nekog člana porodice... takve lozinke se lako pogode, što donosi različite probleme, pre svega onome čije ime je lozinka ukradena. Izabranu lozinku docnije možete da menjate.

Izbor radnih parametara

Ostaje još da odaberete radne parametre (arhiver, protokol i slično) - ne brinite ako ne znate šta je šta, pošto će vam Sezam predložiti uobičajene vrednosti.

Unos ličnih podataka

Prilikom unosa adrese i telefonskog broja budite veoma pažljivi jer netačno unesena adresa onemogućava kontakt Uprave Sezama sa vama, a samim tim i vaše učlanjenje. Unošenjem adrese i ostalih podataka završena je prva faza vašeg prijavljivanja na Sezam.

Redovno

Sledećeg radnog dana biće vam poslato pismo sa obaveštenjima o Sezamu i popunjenom uplatnicom. Ukoliko želite da proverite da li su podaci koje ste uneli ispravni, pozovite sledećeg dana, unesite izabrani pseudonim i lozinku. Sezam će potvrditi da vam je pismo poslato i zahtevati od vas da dopunite podatke. Pošto dobijete pismo, treba da odaberete period na koji se pretplaćujete i na osnovu toga uplatite odgovarajući iznos.

period	pojedinci	firme
1 mesec	10 din	50 din
2 meseca	20 din	100 din
6 meseci	50 din	250 din
12 meseci	80 din	400 din
6 meseci	100 din	500 din

Pojedinci: 90 minuta dnevno, 30 minuta po pozivu
Firme: 480 minuta dnevno, 30 minuta po pozivu

Na uplatnici koju dobijate od redakcije uočene su u redu ispod vašeg korisničkog imena i cene u novim dinarima. Cenu pretplate možete saznati i u Sezamu komandom PRETPLATA, i ako se na dan uplate javite upravi glasom na broj (011) 653 748.

Nakon izvršene uplate, fotokopiju potvrde pošaljite poštom na adresu "Računari za SEZAM", Bulevar Vojvode Mišića 17, 11000 Beograd, ili faksom na brojeve (011) 648 140 i (011) 647 955. Uprava će vam na dan prijema potvrde odobriti pristup sistemu, a sledećeg dana će na vašu adresu biti poslato Uputstvo za korišćenje Sezama, atraktivno štampana knjižica pomoću koje ćete upoznati sve mogućnosti sistema.

...i ekspresno učlanjenje

Oni kojima se žuri mogu postati korisnici Sezama za samo nekoliko sati. Javite se na već opisani način, a onda, ne čekajući na pismo, uplatite odgovarajući iznos na pošti ili banci navodeći sledeće podatke: **Pošiljalac:** vaše ime, prezime, adresa, OBAVEZNO PSEUDONIM koji ste izabrali; **Primalac:** BIGZ Računari; **Svrha uplate:** Pretplata na Sezam; **Žiro-račun:** 40802-603-6-23264; **poziv na broj:** 05 108 66. Potvrdu o uplati pošaljite faksom do 11 časova (radnim danom) - već u popodnevnim časovima ste punopravni korisnik Sezama!

Pretplata na Sezam može se uplatiti i gotovinski u BIGZ-u svakog radnog dana od 9 do 14 časova.

Dvostruko prijavljivanje

Ukoliko tokom predstavljanja imate problema (na primer veza se prekine), pozovite ponovo i prijavite se navodeći kompletne podatke. **Pažnja:** ako ste se prijavili dva ili više puta, koristite ono korisničko ime koje je upisano na uplatnici.

Koliko važi uplatnica

Naše uplatnica važi godinu dana, a pretplatu treba uplatiti po cenama koje **važe na dan uplate**. To **ne** znači da se morate ponovo prijavljivati - jednostavno je da pozovete Sezam, predstavite se izabranim imenom i lozinkom i sistem će opisati ažurne cene pretplate.

Jovan Bulajić

Windows 95 je podigao najviše prašine i među datotekama na Sezamu Tokom oktobra pristigao je veliki broj novih programa od kojih je pozamašan broj bio namenjen baš novom *Microsoft* ovom operativnom sistemu Neki od njih su bili nove 32-bitne verzije ranije već poznatih *shareware* alatki drugi su bili neki sasvim novi dobri programi a veliki broj arhiva koje su u oktobru pristigle na Sezam potekao je i od samog *Microsoft*-a - razni uslužni programi, novi drajveri informativne datoteke i sveže zakrpe (već!) za *Windows 95*

Zakrpe za Windows 95 stigle su od *Microsoft*-a i smeštene na Sezam u w \system direktorijum Ove zakrpe ispravljaju probleme u *Shell32 dll* (shelup exe 390 Kb sitnije izmene vezane za dodavanje novog štampača i ispravljeni bug koji se ponekad javlja kod kopiranja datoteka samih preko sebe - pripazite pošto je nekoliko korisnika prijavilo probleme u radu *WordPad*-a i još nekih programa posle instalacije ove zakrpe) i sigurnosnim mehanizmima pri deljenju datoteka i štampača za *Microsoft* (datoteka vservupd exe, 147 Kb) odnosno *Novell* mreže (nservupd exe 156 Kb)

Od *Microsoft*-a je takođe stigla i lista aplikacija kompatibilnih sa *Windows 95* (w \info\win95app exe 192 Kb) kao i program kojim možete pre instalacije *Windows 95* proveriti da li je vaš hardver odgovarajući (w \util\w95check.zip 61 Kb)

Nove alatke

Među novitetima je i desetak kratkih uslužnih programa u javnom vlasništvu za DOS i *Windows* (pre svega verziju 3 x) sa top liste najkorisnijih programa iz časopisa **PC-Magazine** Skrećemo vam pažnju na program sa vrha ove liste *Find Dead DLLs* pronalazi na disku DLL datoteke koje više ne koristi ni jedan program (w \util\fddlls zip 62 Kb)

Stigle su i nove verzije antivirusnih programa **ThunderBYTE Anti Virus** (TBAV) verzija 6 50 za *DOS* i *Windows 95* (DOS verziju možete naći u r \virus\tbav650 zip a *Windows 95* verziju u w \util\tbw95650 zip prva arhiva je dužine 320 Kb a druga 680 Kb) **F-PROT** v2 20 (samo DOS verzija r \virus\fp-220 zip 590 Kb) i **McAfee ViruScan** (odnosno skraćeno *SCAN*) za *Windows 95* (w \util\scan95 * 1 25 Mb)

Izašla je i nova verzija **WinZip**-a jednog od najtraženijih uslužnih programa za *Windows* Verzija 6 0 ovog *shell* a za lakši rad sa arhivama (pre svega ZIP ali i drugima) izašla je u standardnoj *Windows 3 x* varijanti (w \util\winzip60 exe 320 Kb) i 32-bitnoj verziji za *Windows 95* i *NT* (w \util\winz9560 exe 345 Kb)

WinZip-u možete prepustiti brigu oko arhiva a program koji vam ovog puta preporučujemo (dok se možda ne pojavi neki brži i bolji) za pregled i konverziju slika raznih formata u *Windows 95* je **PolyView** Iako je naš lični favorit za rad sa slikama **Paint Shop Pro** (čija je nova verzija 3 11 takođe stigla u oktobru na Sezam w \graph\psp311 *, nešto više od 2 Mb) prednosti *PolyView*-a su što je kratak i izuzetno brz To naročito dolazi do izražaja kod JPEG (JPG) datoteka kod kojih se ostali programi prilično oznoje pre nego što ih dekodiraju i prikažu (w \graph\polyv215 zip 276 Kb)

Još jedan Power Toy



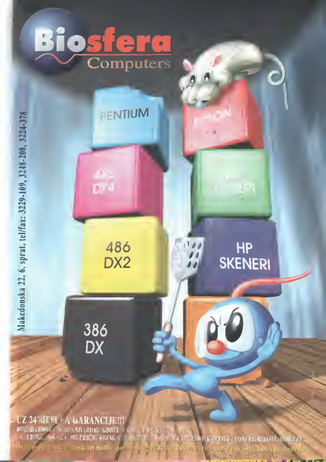


Prošlog meseca smo pisali o *Power Toys* paketu delu ljudi iz razvojnog tima *Windows 95* a ovog puta titulu najboljeg novog programa na Sezamu u oktobru (kad već još uvek nemamo pravu top listu) dodeljujemo novoj moćnoj igrački koja je "poklon" istog tima - programu pod (ne baš inventivnim) nazivom **Tweak UI**

Tweak UI je dodatak za *Control Panel* (gde se instalira kao još jedna ikonica) koji omogućava podešavanje raznih sitnica vezanih pre svega za korisnički interfejs koje se inače ne mogu podešavati (osim korišćenjem *registry* editora što je ne baš pogodan postupak za većinu prosečnih korisnika)

Mogu se menjati brzina otvaranja menija (posle čega se uz zadržavanje miša na njemu otvara kaskadni meni inače povećanje brzine otvaranja je vrlo zgodno za "jurnjavu" po *Start/Programs* podmenijima) osetljivost miša, "animirano" otvaranje i zatvaranje prozora (isključivanje ove opcije je preporučljivo na računarima slabije konfiguracije) promeniti oznaka za prečice (umesto pune strelice koja često potpuno pokvari originalni izgled ikone možete staviti tanju strelicu ili neku drugu ikonu) određivati koje će ikone biti prikazane na radnoj površini (desktop u) kreirati nove *document templates* menjati parametre dizanja sistema (one koji se inače menjaju u msdos sys datoteci) i slično

Aktuelna verzija *Tweak UI* nosi oznaku 0 95 i u pratećoj dokumentaciji se spominje nekoliko potencijalnih problema i grešaka (ah taj *Microsoft*) koji bi trebalo da budu ispravljeni u nekoj sledećoj verziji Bez obzira na to pokupite ovaj program i dodajte ga u vaš *Control Panel* sigurni smo da će vam neka od njegovih opcija zatrebati Pokupite arhivu sa Sezama (w \system\tweakui exe samoraspakujuća ZIP arhiva dužine 61 Kb) raspakujte je u neki privremeni folder kliknite desnim dugmetom iz *Explorer* a na TWEAKUI INF datoteku i izaberite "Install" ■



Biosfera
Computers
Makedonska 22, 6. sprat, tel/fax: 3229-109, 3248-208, 3224-378
PENTIUM
486
DX2
386
DX
HP
SKENERI
UZ 24 MESECA GARANCIJE!!!


ComTrade
Computers
Conner
Canon
Epson
Daewoo
Hewlett Packard
Mitsumi
AT&T
222 41 51
222 26 52
222 41 39
Posao = Zabava

Miljan Mitrović



Delphi

Specijalni dodatak časopisa ***računari***
novembar 1995

Autor se zahvaljuje Borisu Bjeloševiću na pomoći pri izradi ovog umetka

UVOD

Kada je daleke 1983 godine na svet došao *Turbo Pascal* doneo je sa sobom pregršt novotanja integrisani editor i kompajler brzo prevođenje nezavisne izvršne fajlove itd Za "sveže" programere tek stasale na legendarnim računarima kao što su *Spektrum* *Komodor*, *TP* je bio otkrovenje. Lakoća rada koju je *Borland* pružao retko se gde sretala u to vreme linijskih editora i kompajlera iz komandne linije Nedugo zatim počele su da se nižu nove verzije 2.0, 3.0 4.0, 5.0 svaka sa dodatnim poboljšanjima. Ubačeni su moduli EXE fajlovi povezivanje sa C om da bi naposletku (počev od v 5.5) *Borland* ov paskal postao objektno orijentisan što je otvorilo vrata neslućenim mogućnostima

Sve ovo vreme *Borland* je pratio trendove i standarde računarske industrije, što je rezultiralo pojavom paskal kompajlera za *Windows* Međutim neuabilna kakva jeste industrija je postavila nove trendove u *Windows* programiranju pojmence sistem za ubrzani razvoj aplikacija sa oznakom **RAD** (*Rapid Application Development*) Sam trend je započeo *Microsoft* serijom svojih "*visual*" alatki počev od *VisualBASIC* a preko *Visual C/C++*-a pa sve do *VisualFOX*-a Sam *Borland* je suprotno dosadašnjim iskustvima relativno okasnio sa puštanjem svog **RAD** alata Međutim ono što je letos stiglo iz *Borland*-a definitivno je vredelo čekati

Ubica *VisualBASIC* a kako *Delphi* nazivaju iz milošte doneo je kako to samo *Borland* ume da izvede dosta novih stvari U ovom umetku nastojaćemo da predstavimo bar deo tih novina kao i početno uputstvo za instalaciju i konfigurisanje samog *Delphi* ja

Prikazaćemo osr ove dizajniranja aplikacija za koje vam neće biti potrebno odlično poznavanje *Object* paskala bez kojeg u klasičnom paskalu za *Windows* nećete daleko dogurati Bacićemo pogled na nove objektne ekstenzije paskala kao i na neka proširenja samog jezika, što će najviše interesovati naprednije korisnike. Kao "dodatak dodatku" ubacili smo opis stavke menija sa njihovim ekvivalentima na *Speedbar* u, kao i spisak komponenata koje dolaze uz *Delphi*

Kad naposletku budete sei i sami počeli da istražujete, videćete zašto je aktuelna izjava. "Posle pet minuta rada mislio sam da radim sa *VisualBASIC* om, posle petnaest sam znao da je *Delphi* sve ono što bi VB trebalo da bude"

Sve u svemu bon appetit

INSTALACIJA

Hardverski zahtevi

Delphi zahteva minimum 386 ili jaču mašinu (preporučuje se 486) sa instaliranim Windows-om 3.1+ i sa minimum 6Mb RAM-a. Za Client/Server verziju minimum je 8Mb RAM-a, a preporučuje se 12Mb. Jedini nezahtevni deo Delphi-ja je DCC.EXE (DOS kompajler za komandnu liniju) koji traži 1Mb RAM-a. Aplikacije razvijane u Delphi-ju zahtevaju bar 286 mašinu sa minimum 2Mb ukupnog RAM-a i, naravno, sa instaliranim Windows-om 3.1+.

Diskete/CD

Delphi dolazi na CD-u na kojem se nalazi instalaciona verzija zgodno podeljena u četrnaest direktorijuma koji pretpostavljate staju na četrnaest HD disketa. Na prvoj disketi (odnosno u root-u CD-a) nalazi se SETUP.EXE kojim počinjete instalaciju. Ovakva podela (na direktorijume) ne znači da je program moguće instalirati jedino sa disketa, već je postavljena da bi se olakšalo pravljenje backup kopije (što je dozvoljeno po copyright zakonu), ukoliko nastupi oštećenje originalnog medijuma. To, otprilike, obuhvata sve što pruža disketna verzija.

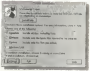
Na CD-u se takođe nalaze sve knjige iz Delphi paketa prelomljene i spremne za čitanje iz Adobe Acrobat Reader programa koji takođe dolazi na CD-u. Iako je li stanje papirne dokumentacije mnogo bolje od elektronske, ovaj potez Borland-a je za svaku pohvalu, naročito ako nekom pozajmite svoje knjige. Sve u svemu, na disku se nalazi ukupno četrnaest (zar opet?) megabajta PDF (Adobe Reader format) fajlova.

Pored ovih stvarčica na CD-u se može naći i kompletna verzija Delphi-ja (spremna za upotrebu bez potrebe za instalacijom) kao i sorsevi nekih delova RTL-a (run time library). Kompletan sors RTL-a i objektnih ekstenzija Delphi-ja ne nalazi se na CD-u, ali se može naći kao poseban paket.

Pošto uz Delphi dolazi distribuciona licenca, uključeni su i BDE (Borland Database Engine), tj. biblioteka rutina za manipulaciju relacionim bazama (koja vam je neophodna ako koristite Delphi za razvoj baza) i ReportSmith run time verzija za štampanje izveštaja. Ove pakete (dve, od nosno pet disketa) morate da instalirate na svim mašinama koje će koristiti vaše aplikacije. Iako Borland preporučuje da se ovi paketi instaliraju u celosti, postoji način da se smanji broj neophodnih fajlova, ali više o tome na kraju.

Postupak instalacije

U zavisnosti od toga da li instalirate sa CD-a ili sa diskete, čeka vas nekoliko, odnosno desetak minuta čekanja. Sama instalacija počinje startovanjem SETUP.EXE programa, nakon čega treba proći kroz nekoliko nivoa instalacionog programa.



Slika 2.1

1) Na početnom ekranu (slika 2.1) birate između kompletne (Complete), minimalne (Minimum) ili korisnički definisane (Custom) instalacije. Instalacioni program predlaže kompletnu instalaciju, međutim imajte u vidu da ona zauzima oko 80Mb na disku što i nije malo. Sa druge strane, minimalna instalacija (28Mb) prebacuje samo osnovne fajlove za rad samog Delphi-ja i pod takvom instalacijom možete da zaboravite na izveštaje i mrežne aplikacije. Kompromis između ove dve opcije je Custom instalacija gde vi birate ono što vama treba. Tabela 2.1 vam pruža pregled svih delova Delphi-ja sa njihovim veličinama kako biste lakše odlučili šta želite da instalirate. Nakon izbora tipa instalacije pritisnite dugme Continue.

Napomena. Osim ove tri stavke, prvi dijalog prikazuje i polje sa nazivom Windows LAN. Ovo polje morate da uključite ukoliko instalirate Delphi na lokalni disk, a koristite Windows sa read only mrežnog diska. Kada je ova opcija uključena, instalacioni program će kreirati direktorijum DELPHI\WINSYS na lokalnom disku i u njega smestiti sve fajlove koje bi inače smeo u Windows ov SYSTEM direktorijum (a sada to ne može da uvede pošto nema pristup za pisanje). Pre nego što budete u mogućnosti da startujete Delphi, morate obavestiti administratora mreže da prebaci sve fajlove iz WINSYS direktorijuma u SYSTEM direktorijum Windows-a (pošto on sigurno ima pristup tom direktorijumu).

Tabela 2.1

Delphi	**49 056Kb**
izvršni fajlovi	21 856
Primeri (Samples)	15 936
Komandna linija (alati)	1 424
Korisnički help	1 168
Tutori	8 016
RTL sorsovi	656
ReportSmith	**29,088Kb**
ReportSmith	16 432
RunTime (veza sa BD)	9 344
Primeri (Samples)	928
Veze sa bazama,	2 384
dBase	608
Paradox	368
Text	176
Excel	144
FoxPro	608
Btrieve	176
Access	304
InterBase Server	**6 448Kb**
Database Desktop	**2,928Kb**
Winsight/Winspector	**832Kb**
Borland Database Engine	**4 624Kb**
Ukupno:	**84,704Kb**

2) Bez obzira na tip instalacije koji ste izabrali Delphi vam pruža priliku da promenite imena direktorijuma u koje će izabrani delovi biti instalirani. Mi vam predlažemo da ukoliko ne znate šta je šta sve ostavite kako jeste. Ukoliko ste izabrali Custom instalaciju pruža vam se prilika da osim direktorijuma promenite i spisak delova koji će biti instalirani. Ispred svih stavki pojaviće se polja sa kvačicama (CheckBoxes) koje možete uključiti/isključiti i time izabrati odnosno izbaciti određene delove Delphi-ja. Pošto ste sve namestili kako vam odgovara, izaberite Continue.

3) Ukoliko ste odlučili da instalirate lokalni InterBase server pojaviće se dijalog za izbor protokola koje će InterBase koristiti za pristup drugim mrežnim serverima. Ponuđeni su NetBEUI protokol (Windows mreža), IPX/SPX protokol (Novell NetWare mreža) i TCP/IP protokol (njega uspešno podržavaju skoro svi mrežni softveri). Prva dva se ne konfigurišu, dok se za TCP/IP mora navesti tip TCP/IP protokola (kako bi se instalirao adekvatni WINSOCK.DLL) i položaj SERVICES fajla (kako bi se InterBase registrovao). Ukoliko niste uvezani u neku specijalnu mrežu (ili uopšte niste uvezani) ostavite samo NetBEUI koji ionako ne može da se isključi. Ovaj dijalog se ne pojavljuje pri minimalnoj instalaciji.

4) Ukoliko ste izabrali bilo koji od delova Delphi-ja koji koristi pristup bazama, automatski će se instalirati BDE (Borland Database Engine). U tom slučaju pojaviće se dijalog sa dva polja gde možete uneti nazive direktorijuma u koje će biti instalirani izvršni fajlovi i jezičke tabele sa jedne strane i konfiguracioni fajl sa druge strane. Kao i u prethodnim slučajevima preporučujemo da sve ostavite kako jeste i izaberete Continue.

Posle svih ovih perpetija, ukoliko ste sve dobro uneli pojaviće se finalni prozor sa "velikim" malim slovom i i informacijom da ste spremni za instaliranje. Sve što treba da uradite je da izaberete dugme Install i instalacioni program će iskopirati sve neophodne fajlove na njihova odredišta, namestiti INI fajlove i prikazati vam klasičan README.TXT fajl sa informacijama koje su pristigle nakon pripreme knjiga te se u njima nisu mogle naći. Ovim je instalacija Delphi-ja završena.

Startovanje

Po završetku instalacije u Program Manager-u (ili Dashboard-u šta već koristite) biće otvorena nova grupa pod konspirativnim imenom Delphi ("ajde") kao što je ona na slici 2.2. Grupa će sadržati nekih dvadesetak ikonica od kojih većinu nikad nećete direktno koristiti (radi

se o helpovima koje ćete najčešće aktivirati (u se programa). Za početak najvažnija koncepcija je ona pod koje piše Delphi, zato aktivirajte tu aplikaciju dvostrukim klikom i uđite u magični svet Delphija.



Slika 2.2

ELEMENTI PROGRAMA

Vizuelni elementi

Po startovanju programa na ekranu se pojavljuju četiri početna prozora, kao na slici 3.1. Za razliku od Visual-BASIC-a koji ima providni radni deo (client-area), Delphijev glavni prozor ga nema.



Slika 3.1

Iz samog glavnog prozora mogu se birati opcije iz menija, sa speedbar-a, kao i komponente iz palete istih. Sam dizajn obavlja se u nezavisnom prozoru (Delphi nije klasična MDI aplikacija) koji na početku nosi oznaku Form1. Pored ta dva prozora Delphi otvara i Object Inspector prozor iz kojeg se komponente (delovi aplikacije) mogu menjati i prilagođavati zahtevima programera. Sve vizuelne promene nad aplikacijom (njenim formularima, Forms) automatski se registruju u odgovarajućim sors fajlovima (na početku se otvara jedan, zajedno sa formularom) i skrivenim, najčešće DFM fajlovima, čime se u mnogome olakšava razvoj, pošto se programer pošteđuje silnog kucanja raznih rezervisanih reči.

Glavni prozor

Glavni prozor se sastoji iz tri dela: menija, SpeedBar-a i palete komponenti (slika 3.2). Uloga mu je da koordinira rad Object Inspector-a i raznih formulara i njihovih sors fajlova, da obezbeđuje kompajliranje i izvršavanje aplikacija kao i integrisano debagiranje.

Meni je podeljen u više podmenija, shodno različitim operacijama. Na kraju dodatka imate spisak svih delova menija (sem Tools) sa njihovim ekvivalentnim dugmićima za SpeedBar. Pritiskom na neku opciju iz SpeedBar-a dajete nalog programu da uradi istu onu stvar koju bi uradio da ste izabrali neku stavku iz menija. Jedina prednost koju SpeedBar ima nad n[...] određenoj opciji (P[...]

Pri prvom startovanju Delphi kreira [...] SpeedBar sa nekoliko opcija vezanih za otva[...] formulara i projekata, kao i dugmiće za prevođenje i [...]tovanje vaših aplikacija. Pošto će vam kasnije (kada počnete da radite sa Delphijem) neke opcije biti potrebnije od drugih i poželećete da ih imate na SpeedBar-u, u ovaj dodatak smo ubacili opis načina na koji možete da konfigurišete svoj SpeedBar. Početni izgled SpeedBar-a (kakvim ga vidite prvi put) predstavlja osnovni (default)

Slika 3.2



zgled. Za sada je važno da to zapamtite zbog jedne opcije koja se javlja pri konfigurisanju (o čemu će biti reči kasnije u dodatku).

Paleta komponenata je deo glavnog prozora koji ćete najčešće koristiti pošto sa nje "skidate" neophodne komponente i ubacujete ih u svoje aplikacije. Shodno postojanju različitih grupacija komponenata (klasične kontrole, komponente za pristup bazama, 3rd Party komponente...) paleta je podeljena na više stranica (čija imena možete pročitati na tabovima ispod sličica komponenata). Paletu je moguće proširivati kako novim komponentama, tako i novim stranicama (ukoliko, na primer, ubacujete dosta novih komponenata koje imaju nešto zajedničko).

Object Inspector

Kao što je paleta komponenata najčešće korišćeni deo glavnog prozora, tako je u celom Delphi-ju ovo najkorišćeniji prozor. Sva moguća podešavanja formulara i komponenata obavljaju se baš ovde. Veličina, pozicija, stil, fontovi, boje itd. sve se to menja u ovom prozoru. Da li je promena nastupila na samom formularu (npr. mišem ste pomerili komponentu) ili je vrednost unesena ručno, uopšte nije važno, pošto se sve registruje automatski i prikazuje se ispravna vrednost. Sve ovo, naravno, važi za prvu stranicu ovog prozora (stranice menjate klikom miša na ime koje se nalazi pri dnu prozora) sa nazivom Properties.

Druga stranica (Events) obuhvata sve događaje koje određena komponenta (ili formular) može da generiše. Na početku su sva polja ove stranice prazna, međutim, kako razvijate aplikaciju tako će se ova polja popunjavati. Nemojte misliti da sva ova polja moraju da se popune. Ne moraju, niti ćete čak doći u situaciju da vam bude neophodno da popunite sva polja. Komponenta u tom slučaju (prazno polje za neki događaj) jednostavno ne generiše ništa i događaj biva ignorisan. Sa druge strane, ovakav sistem (preko događaja) je jedini način da komponente obave neki posao koji vama treba, pošto je većina (naročito vizuel-

[illegible] uznom Events stranicom jed[illegible] na ekranu.

Ukoliko vam se desi da greškom ili namerno ugasite Object Inspector prozor, možete ga ponovo aktivirati iz menija preko **View|Object Inspector** ili pritiskom na F11. Object Inspector sam nestaje sa ekrana samo u dva slučaja: pri izlasku iz programa i pri startovanju vaših aplikacija iz samog Delphi-ja. Po povratku iz vašeg programa prozor se vraća sam.

Formulari i sorsevi

Sve poslove vezane za dizajn najlakše ćete obaviti iz Form prozora. Miš je, naravno, nezamenljiv u ovakvim situacijama i njega ćete sigurno koristiti (može i preko tastature, ali je pravo mučenje). Dodavanje komponenata, njihovo raspoređivanje, finalna grafička dorada samo su neki od poslova koje ćete obavljati u ovom prozoru. Sam prozor je u tesnoj vezi sa Object Inspector-om i sve promene u jednom automatski se registruju u drugom, što održava konzistentnost podataka i oslobađa vas obaveze da sve kucate dvaput (što bi teško prolazilo bez grešaka).

Iza formulara se nalazi sors prozor u kojem se nalazi telo programa (modula). Postoji samo jedan prozor ovog tipa koji, međutim, može sadržati više stranica od kojih svaka predstavlja poseban sors. Ukoliko menjate sors, bilo ručno bilo preko formulara, potrebno je da ga kasnije snimite. Ukoliko pokušate da izađete iz Delphi-ja, dobićete poslednju mogućnost da program snimite na disk.

Pod konfigurisanjem se podrazumeva podešavanje integrisane okoline i njenih pratećih elemenata kako bi nam rad u Delphi-ju bio prijatniji i brži. Većina elemenata Delphi-ja može se konfigurisati izborom komande **Configure** iz lokalnog menija (lokalni meni dobijate pritiskom na desno dugme miša kada je kurzor iznad željenog elementa). Najvažnija stvar koju odmah treba da konfigurišete je radna okolina. To radite izborom komande **Options|Environment** koja će otvoriti dijalog za konfigurisanje okoline.

KONFIGURISANJE

Konfigurisanje okoline

Dijalog se sastoji iz sedam stranica (ovakva podela na stranice je česta pojava u Delphi-ju) koje birate mišem pritiskom na levo dugme kada se kurzor nalazi iznad naziva stranice. Nazivi se nalaze tik ispod samih stranica i karakteriše ih specijalan izgled (nalik na jezičke na fasciklama). Počećemo od prve, koja se bavi opštim stvarima.

Preferences

Na ovoj stranici (slika 4.1) postoji dosta opcija, prva od njih pod nazivom Desktop contents, odnosno sadržaj desktop-a. Ova opcija određuje koje će informacije biti snimljene u desktop fajl (Delphi ovaj fajl koristi da rasporedi prozore na ekranu u onaj poredak kakav je postojao kad ste snimili fajl, omogućavajući vam time da prosto "nastavite" gde ste stali). Desktop only znači da će u fajl biti snimljen samo raspored prozora. U suprotnom (Desktop & symbols) u fajl će biti ubačeni i simboli aplikacije. Prvo rešenje oslobađa prostor na disku, drugo ubrzava rad, pošto nije potrebno ponovo prevoditi program da bi se stvorila nova tablica simbola (procedure, funkcije, varijable, objekti itd.). Ovu tablicu koristi Browser (prozor za vizuelni pregled simbola programa) te ako vam on ne treba, isključite ovu opciju.

Druga grupa opcija koristi se za uključivanje (i isključivanje) automatskog snimanja. Ponuđene su dve stvari koje se mogu automatski snimati: sors fajlovi u editoru i sadržaj desktop-a. Ukoliko je neka od ovih opcija uključena, kad god budete startovali aplikaciju iz Delphi-ja, ti fajlovi će automatski biti snimljeni. Pazite da se u tom slučaju ne prevarite i da vam fajlovi budu snimljeni iako vam to nije bila namera. U tom slučaju originalni sadržaj možete "pokupiti" iz backup fajla.

Form Design grupa opcija kontroliše prikaz i veličinu mreže (grid) tačkica koje vam olakšavaju rad pri dizajniranju. Poenta je u tome što se sve komponente kolje stavite na formular lepe za najbližu tačku, što vam omogućava da, na primer, poređate nekoliko komponenti jednu iznad druge ali u stojinju po vertikali. T[illegible] kođe, kada pomerate postojeću komponentu po formularu, Delphi vam neće dozvoliti da je smestite u [illegible] između tačaka (tačnije, samo gornji levi ugao s[illegible] tačke, pozicija ostalih uglova zavisi od veličine same komponente). Početna veličina mreže je 8x8 piksela i unesena je u polja Grid Size X i Grid Size Y.



Slika 4.1

Prekidač Display Grid određuje da li će se na površini formulara videti mreža (preporučujemo da ostavite ovo polje uključeno, kako biste lakše razlikovali dizajn verziju formulara od izvršne, koju dobijete kad startujete vaš program). Poslednja opcija, Snap to grid, kazuje Delphi-ju da li da "lepi" komponente za mrežu. Ukoliko je ova opcija isključena, ostale opcije iz ove grupe imaju malo efekta. Mi vam predlažemo da koristite mrežu pri dizajniranju, a da u slučaju potrebe za sitnim pomeranjima izmene izvršite u Object Inspector-u. Skoro sve komponente imaju properije Top i Left koji određuju gornji levi ugao i properije Width i Height koji predsta-

vlaju širinu i visinu komponente. Pri ovakvom ručnom pomeranju komponenti mreža nema nikakvog uticaja.

Četvrta grupa opcija, pod nazivom *Debugging*, odnosi se, naravno, na opcije integrisanog debagera. Prva od tih opcija, *Integrated debugging*, određuje da li je integrisani debager uopšte uključen. Ukoliko nije, onda naredne dve opcije nemaju nikakvog efekta. Prva od tih, *Break on exception*, omogućuje *JustInTime* debagiranje koje vam olakšava posao utoliko što pri nastanku nekog prekida (greške) program prestaje sa radom, obaveštava vas o klasi greške i poruci, a zatim startuje integrisani debager na prvoj sledećoj poziciji za koju postoje debag informacije. Time ste oslobođeni "jurcanja" po programu i traženja grešaka – jednostavno sačekate da se greška javi sama. Sledeća opcija je *Step program block* koja vam omogućava da debagirate i početak one delove modula uključene u aplikaciju, naravno pod uslovom da za taj modul postoje debag informacije.

Preostale dve opcije rade bez obzira na stanje integrisanog debagera i definišu ponašanje *Delphi*-ja pri startovanju vaših aplikacija. *Minimize on Run* smanjuje (minimizuje) sam *Delphi* i time oslobađa ceo ekran za vašu aplikaciju, dok druga (*Hide Designers*) skriva samo *Object Inspector* i dizajn prozor. Nevolja sa minimizacijom *Delphi*-ja je u tome što se, ukoliko integrisano debagiranje nije uključeno, po završetku rada vaše aplikacije *Delphi* ne vraća u prvobitno stanje. Jednostavno, nije obavešten da se aplikacija završila, i ostao je dole. U tom slučaju morate sami da ga vratite dvostrukim klikom na ikonicu.

Preostale dve grupe ikonica (*Compile* i *Gallery*) određuju da li će se pri prevođenju aplikacija i modula pojavljivati prozorčić sa statusom (uključite ovo polje za probu), odnosno da li će se pri kreiranju novih aplikacija i formulara korisniku dati da izabere iz galerije već urađenih. Te galerije možete kasnije dograđivati svojim aplikacijama i formularima.

Konfigurisanje biblioteke komponenata

Sve komponente iz palete komponenata nalaze se u jednom jedinom fajlu koji nosi ekstenziju **DCL** (*Delphi Component Library*). Ova stranica (*Library*) definiše opcije pod kojima se rekompajliraju biblioteke (može ih biti više, tačnije za svaki projekat možete imati po jednu). Dakle, sve ove opcije neće imati nikakvog efekta dok ne izaberete **Options|Rebuild Library** komandu iz menija. Ta komanda će uzeti u obzir ove opcije i pod njima ponovo prevesti sve module čije se komponente nalaze u paleti. O prve dve grupe opcija (*map file* i *Link buffer*) biće više reči u petom poglavlju u opcijama *Project Manager*-a. Za sada ćemo se zadržati na trećoj grupi (*Options*) u kojoj se nalaze dve zanimljive opcije.

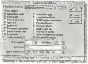
Ukoliko želite da imate mogućnost da prilikom debagiranja ulazite u sors kodove komponenata, morate uključiti opciju *Compile with debug info*, koja će pri prevođenju modula u biblioteku uključiti debag informacije. Naravno, ovo sve važi samo za module koje dobijete u sors kodu. Za fabričke module (tj. module nezavisnih proizvođača) koji dolaze samo u prevedenoj formi (bez sors fajla), ova opcija nema nikakvog efekta (kao uostalom ni većina opcija na ovoj strani).

Druga opcija (*Save library source code*) pri rekompajliranju biblioteke kreira paskal fajl koji predstavlja sors kod biblioteke. Od ovog fajla nećete imati ama baš nikakve praktične koristi sem tekstualnog pregleda svih modula i njihovih registracionih poziva.

Poslednje polje ove stranice predstavlja listu direktorijuma u kojima će *Delphi* tražiti sors fajlove modula. Ukoliko negde ne nađe sors fajl, *Delphi* će smatrati da ga modul ni nema i neće ga rekompajlirati.

Opcije editora

Na ovoj strani (slika 4.2) nalaze se opcije vezane za integrisani editor *Delphi*-ja. Prva od njih, *Editor SpeedSetting*, određuje koje će akceleratore (kombinacije tastera koje automatski obavljaju neki posao) editor prepoznavati. U početku je tu osnovni raspored, ali ga možete menjati u neki drugi (klasičan **IDE**, kao onaj iz **BP70**, **EPSILON** ili **BRIEF** emulacija), sve u zavisnosti od okoline u kojoj ste naučili da kucate. Druga grupa opcija sadrži nekih dvadesetak prekidača koji definišu ponašanje editora pri kucanju.



Slika 4.2

Ispod ove glomazne grupe opcija nalazi se nekoliko polja sa zanimljivim funkcijama. *Undo limit* predstavlja veličinu bafera za undo komandu i postavljen je na 32k. Ukoliko malo grešite, možete slobodno da smanjite ovu vrednost (memorije nikad dosta). Kontrola tabulatora (*Tab stops*) definiše horizontalne pozicije na kojima će se kurzor zaustavlti pri tabulaciji. Ukoliko je uključena *SmartTAB* opcija, ove vrednosti se uzimaju u obzir samo u prvom redu i u redu iznad kojeg se nalazi bar jedan prazan red.

Sintaksne ekstenzije (poslednje polje) sadrže spisak svih ekstenzija (odvojenih znakom ; (tačka zapeta)) koje će editor "farbati", tj. prikazati u različitim bojama. Radi se o postavci fajlova koji imaju tačno definisanu sintaksu, rezervisane reči itd. U editoru ćete moći da editujete bilo koji tekstualni fajl, ali će samo oni čija se ekstenzija nalazi u ovom polju biti višebojni.

Izgled editora

Sledeća strana na listi, *Editor Display* (slika 4.3), sadrži polja koja definišu izgled editora. Prva od tih grupa sadrži četiri opcije: izgled kurzora (**BRIEF** kurzor je horizontalan, inače je uspravan), kreiranje **BACKUP** fajlova (*Create backup files*, preporučljivo), očuvanje kraja linije i vrsta zumiranja (kad je ova opcija uključena, sors prozor se može maksimizovati preko celog ekrana, u suprotnom nikad neće pokriti *Delphi* prozor).

Druga grupa opcija je identična izboru vrste editora sa prethodne strane: dakle, birate između klasičnog *Windows* režima, klasičnog *Borland* režima, **BRIEF** ili **EPSILON** editora. Odmah pored ove grupe nalazi se druga koja se bavi desnom marginom i ima dva polja. Prvo, *Visible right margin*, predstavlja vidljivost margine i, ako je uključeno, u sors prozoru će se pojaviti uspravna linija sive boje koja će vas upozoravati na prekoračenje. Postojanje te margine vas ne ograničava u kucanju, jednostavno predstavlja upozorenje da vam nije sve stalo u u željenu liniju. Ovaj podatak vam može biti važan ukoliko nameravate da štampate svoje sors fajlove, pa je potrebno da podesite marginu teksta sa marginom štampača. Ispod tog prekidača nalazi se polje u koje unosite kolika je ta vaša željena margina.



Slika 4.3

Ostatak stranice ispunjen je poljima za izbor fonta i veličine. Ukoliko vam je čudno što u spisku fontova nema stotinjak vaših lepih fontova koje ste mukom prikupili za svoj *Windows*, treba da znate da *Delphi* u editoru koristi isključivo monotype (ne TT) fontove kojih ima veoma malo, i od kojih jedino *Courier* liči na nešto. Pokušajte da nađete negde ekranske fontove za *Windows* 2.0 ili 3.0 i instalirajte njih.

Boje editora

Ova stranica je u potpunosti posvećena menjanju boja kojom se tekst ispisuje u prozoru. Ukoliko ste izbacili sintaksne ekstenzije iz opcije editora, većina ovih boja neće imati nikakav efekat. Sa leve strane prozora imate spisak različitih elemenata teksta koji se može farbati, pored njega tabelu sa šesnaest boja, i na kraju tip slova (bold, italic, underline). Ispod imate prozor u kojem se nalaze sve moguće varijante teksta.

Paleta i Browser

Preostale dve strane ovog dijaloga sadrže opcije za manipulaciju paletom komponenata (dodavanje novih stranica, izbacivanje praznih, brisanje komponenata iz liste) i opcijama browser-a, alatke kojom vizuelno prikazujete hijerarhiju objekata, uključujući i sve nazive polja i metoda. Stranicu za konfigurisanje palete možete takođe pozvati i izborom **Configure** komande iz lokalnog menija palete (desno dugme miša).

Konfigurisanje Speedbar-a

SpeedBar (deo glavnog prozora na kojem se nalaze mini ikone) konfiguriše se iz lokalnog menija (desno dugme miša). Izaberete **Configure** iz tog menija i otvoriće se novi dijalog sa nazivom *SpeedBar Editor*, u kojem se nalaze dve liste: jedna sa spiskom menija, druga sa komandama iz izabranog menija i ekvivalentnom ikonom.

Veličinu *Speedbar*-a menjate tako što postavite kurzor miša između njega i palete komponenata i sačekate da se oblik kurzora promeni. Pritisnete levo dugme miša i razvučete (skupite) *Speedbar* na veličinu koja vam odgovara. Nove ikonice dodajete tako što ih "uhvatite" iz *SpeedBar Editor*-a i prenesete na sam *Speedbar*. Pomeranje u suprotnom smeru ide malo drugačije, tačnije, da biste ikonicu skinuli sa *SpeedBar*-a ne morate da je prebacite nazad u editor, dovoljno je da je iznesete van površine *SpeedBar*-a. U dnu *SpeedBar* editora imate dugme sa nazivom *Reset Defaults*, koje vraća početno stanje, onakvo kakvo je bilo pri prvom startovanju *Delphi*-ja.

Napomena. Na kraju ovog umetka imate spisak svih komandi koje poseduju *Speedbar* ikonice, zajedno sa kraćim opisom poslova koje obavlja.

RAZVOJ APLIKACIJA

Struktura aplikacije

Sve aplikacije razvijene pod Delphi-jem imaju manje-više istu strukturu prikazanu na slici 5.1. U tom primeru se nalazi aplikacija sa tri formulara, od kojih je dan ima tri komponente. Slika je namerno tako napravljena da bi se lepo videla struktura a ne maksimalan broj elemenata. Broj formulara po aplikaciji je ograničen na 32767, kao i broj komponenti po formularu, što je brojka koju ćete teško dostići. Svaki formular i komponente na njemu imaju svoje propertije i događaje koji stoje zapisani u **DFM** fajlu tog formulara, a koji se mogu menjati iz *Object Inspector*-a. Moguće je iz jednog formulara pozvati drugi, kao i razmenjivati poruke i podatke između više otvorenih formulara. Formulari su, onako površinski posmatrano, komponente aplikacije, kao što su npr. dugmići komponente formulara. Tokom razvoja nikad nećete videti tu pripadnost formulara aplikaciji na ekranu, međutim veza postoji.

Kad god krećete sa novim projektom, a niste zabrali da vam se prikaže galerija aplikacija, krećete od blank (praznog) formulara i njegovog sors fajla. Sors aplikacije se ne vidi, ali ga možete u svakom trenutku pozvati preko meni komande **View|Project Source**. Malo je verovatno da ćete ikada koristiti ovaj sors (ako vizuelno dizajnirate celu aplikaciju), te je stoga skriven. Još jedna korisna stvar vezana za projekte je *Project Manager* (slika 5.2) koji pozivate sa **View|Project Manager**. Preko njega manipulišete formularima u projektima (dodajete ili oduzimate) i postavljate sve moguće opcije vezane za projekat (vidi dalje u tekstu).

Postoji nekoliko osnovnih tipova formulara koje *Delphi* podržava, pomenće: Normalni formular, **MDI** Okvir (*Frame*), **MDI** Potomak (*Child*) i formular koji je stalno vidljiv (*StayOnTop*). Tip formulara možete menjati u **ObjectInspector|Properties|FormStyle** polju. Sa druge strane, osnovnih komponenti ima mnogo više, što je i normalno.



Slika 5.1

Pošto je struktura aplikacije izrazito hijerarhijska, imena metoda koje upišete u **ObjectInspector|Events** stranicu automatski se kreiraju kao **metode** u definiciji objekta formulara. Čak se i obrade događaja koje generiše formular nalaze u tom istom formularu, što na prvi pogled izgleda malo čudno. Postavlja se pitanje čemu to, kada je u pitanju ista klasa, pa bi bilo lakše da se odmah uvede obrada događaja umesto preko events stranice. Odgovor se krije u činjenici da je *TForm* (klasa koju nasleđuju svi formulari) generička klasa koja sama po sebi ne obrađuje nikakve događaje, kako i treba da bude, jer se ne zna čemu će sve taj formular koristiti. Mnogo je jednostavnije neke događaje tretirati kao take, nego kao virtuelne metode, i terati korisnika da ih nasleđuje (jako poziv virtuelnog metoda traje kraće od poziva event-hendlera).



Slika 5.2

Hello World

Najprostiji program tipa *Hello World* moguće je napraviti u *Delphi*-ju bez jedne ukucane linije koda. Postupak je veoma jednostavan.

Startujete *Delphi*, sačekate da se otvori novi projekat. Otvoriće se, kao i uvek, jedan formular i njegov sors fajl, i tada na scenu stupate vi. Izaberite na *Standard* stranici palete komponenata **Label** komponentu (ima ikonicu u obliku velikog slova A). Potom kliknite mišem negde u prostor formulara (najbolje negde u sredini) i pojaviće se novi objekat sa nazivom *Label1*. Time ste na formular stavili sve objekte koji su vam potrebni za ovaj program.

Izaberite formular (kliknite mišem negde na prazan prostor unutar formulara ili izaberite **Form1: TForm** sa drop-down liste koja se nalazi u Object Inspector-u tik ispod naslova), izaberite properties stranicu (ako već nije izabrana) i potom kliknite na polje sa nazivom *Caption*. Kurzor će se pojaviti pored sadržaja polja koje glasi *Form1* i omogućiti vam da ga menjate. Za početak obrišite sadržaj polja i unesite **Hello World Aplikacija**. Dok to budete ukucavali, primetićete da se automatski menja naziv dizajn prozora u ono što ukucavate.

Izaberite sada **Label** komponentu koju ste stavili na formular i uradite istu stvar kao i sa samim formularom: promenite polje *Caption*, ali ovog puta u **Hello World**. Time ste i zvanično završili ovu "aplikaciju". Sve što treba da uradite da biste startovali program je da pritisnete **F9** ili odaberete meni komandu **Run|Run**. Program će se prevesti i startovati.

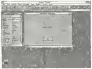
Pošto će se startovani program standardno pojavljivati na položaju koji mu je zadat pri dizajnu, najočigledniji pokazatelj da se program startovao biće u nedostatku grida (ukoliko ga već niste ukinuli) odnosno nevizuelnih komponenata. Takođe će nestati *Object Inspector* i dizajn prozor. Sors prozor ostaje otvoren, što je veoma korisno, jer nam može biti potreban za debagiranje.

Zatvorite aplikaciju sa **Alt-F4** ili biranjem **Close** komande iz sistemskog menija. Po završetku rada programa sve će se vratiti na početno stanje.

Upotreba events stranice

Da biste prikazali kako se koriste događaji pod *Delphi*-jem, na formular (i dalje razvijamo **Hello World Aplikaciju**) ćemo staviti jedno dugmence. Ono se takođe nalazi na *Standard* stranici palete i liči na kvadratić u kojem piše *OK*. Izaberite ga na paleti i postavite na formular ispod teksta *Hello World*. Pojaviće se dugmence sa nazivom *Button1*, što je posledica Delphi-jevog sistema imenovanja novih instanci komponenata. Ime se dobija spajanjem originalnog imena komponente (na paleti) sa prvim slobodnim brojem, što znači da će sledeće dugme koje dodate dobiti ime *Button2* (naravno, samo u slučaju da niste preimenovali prvo).

Promenite ime dugmeta (*Name* polje, properties stranica) u *OkButton*. Ovu izmenu smo ubacili zbog jedne sitnice koja će vam možda smetati u početku, pošto je važan redosled unošenja novih podataka. Kad promenite ime komponente (*Name*), automatski se menja i naslov na ekranu (*Caption*), mada obrnuto ne važi. Dakle, uvek prvo promenite ime (ukoliko to uopšte želite da uradite), a zatim naslov. Važno je zapamtiti da sva imena moraju biti jedinstvena, dok više komponenti može imati isti naslov. U našem primeru promenite naslov u *OK*, a zatim promenite *Default* polje u *True* (default dugme je ono koje se aktivira pritiskom na taster *Enter*, čak i onda kad fokus nije na njemu). Ekran bi trebalo da izgleda otprilike kao na slici 5.3.



Slika 5.3

Ukoliko sad startujete program sa **F9** i potom kliknete mišem na dugme *OK*, neće se desiti ništa. Program, tačnije *OK* dugme, ne zna šta treba da uradi ako ga neko izabere. Za te stvari se koristi events stranica

ObjectInspector a te da biste došli do nje prvo morate da isključite aktivnu aplikaciju (**Alt + F4**).

Izaberite *ObjectInspector* prozor i na njemu *events* stranicu, potom na dizajn prozoru izaberite dugme. Pojaviće se spisak od nekih dvanaest događaja koje dugme može da generiše. Vas za početak interesuje samo prvi nazvan *OnClick* (grubo prevedeno, to znači "na pritisak"). Pritisnite mišem dvaput (double click) na prazno polje pored *OnClick*. Fokus će preći na sors fajl u kojem će se automatski kreirati nova metoda nazvana *OKButtonClick*. Ukoliko prepustite *Delphi*-ju da imenuje metodu (kao što ste sad uradili), dobićete ime komponente povezano sa imenom događaja, što je sasvim dovoljno da se snađete šta koja metoda radi. Umesto toga mogli ste sami da unesete ime metode u polje *OnClick*, i *Delphi* bi prihvatio to ime umesto svog.

Svejedno koje ste ime izabrali, u sors kod unesite samo jednu liniju u kojoj se nalazi poziv metode *Close* (slika 5.4). Pošto metod *OkButtonClick* pripada formularu (deklarisan je kao *TForm1.OkButtonClick*), ova komanda će pozvati *Close* metod formulara koji bi trebalo da ga zatvori. Pošto aplikacija ima samo jedan formular koji je ujedno i glavni, njegovo zatvaranje predstavlja zatvaranje same aplikacije. Startujte program sa **F9** i proverite da li pritisak na dugme *OK* zatvara aplikaciju.



Slika 5.4

Šta se u stvari desilo? Pritiskom na dugme, komponenta dugmeta proverava da li je nešto upisano u *OnClick* polje i ako jeste (u slučaju ove aplikacije) poziva metod koji je upisan (u ovom slučaju *TForm1.OKButtonClick*). Metod koji se pozove može da obavi koju god radnju vi zamislite, pošto niste ograničeni na jednu komandu. U ovom slučaju, međutim, ograničili smo se na zatvaranje formulara (aplikacije).

Sors fajlovi

Za razliku od klasičnog paskala, u *Delphi*-ju postoji više vrsta sors fajlova. Najvažniji sors fajl je **DPR** (*Delphi PRoject*). Što u stvari predstavlja standardni paskal fajl u kojem se nalazi aplikacija. Njega inicijalno ne vidite, ali ga u svakom trenutku možete pozvati sa **View|Project Source**. Retko ćete imati potrebe da menjate ovaj fajl, pošto se u njemu nalazi samo spisak formulara (u uses sekciji čime se obezbeđuje njihovo uključivanje u **EXE** fajl) i komande za njihovo kreiranje. *Delphi* sam održava ovaj fajl u ispravnom stanju, a u njega biste eventualno mogli da ubacite neke komande koje želite da se izvrše pre i nakon završetka rada same aplikacije. Na listingu 5.1 možete da vidite sors fajl *Hello World* aplikacije iz prethodnog poglavlja.

Listing 5.1

```
unit Unit1;
interface

uses
  SysUtils, WinTypes, WinProcs,
Messages, Classes, Graphics, Controls,
  Forms, Dialogs, StdCtrls;

type
  TForm1 = class(TForm)
    Label1: TLabel;
    OKButton: TButton;
    procedure OKButtonClick(Sender:
TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
implementation

{$R *.DFM}

procedure TForm1.OKButtonClick(Sender:
TObject);
begin
   Close;
end;

end.
```

Drugi po važnosti su **PAS** i **DFM** fajlovi u kojima su smešteni sors fajlovi modula koji će biti uključeni u program. Svaki formular ima oba sors fajla, dok neki moduli (npr. neka biblioteka rutina) nemaju **DFM** fajl.

Sors fajlovi **PAS** tipa su standardni tekst fajlovi u **ASCII** kodu i pisani su po sintaksi standardnih modula sa *interface* i *implementation* delovima. U njima su zapisane definicije objekata formulara zajedno sa svim komponentama koje se u njemu nalaze (na listingu 5.2 nalazi se paskal fajl glavnog formulara *Hello World* aplikacije). U ovom fajlu nisu zapisane početne karakteristike komponenata (one unesene na *property* stranici *ObjectInspectora*) već samo njihova paskal imena (ona iz *Name* propertija) i tip komponente.

Listing 5.2

```
program Project1;

uses
  Forms,
  Unit1 in 'UNIT1.PAS' {Form1};

{$R *.RES}

begin
  Application.CreateForm(TForm1,
Form1);
  Application.Run;
end.
```

Borland preporučuje da se nove komponente ne dodaju u ovaj sors fajl nego sa palete, pošto je i slaža ju "ručnog" ubacivanja komponente dodatno komplikovano i kreirati ručno, pa potom inicijalizovati i eventualne propertije (za komponente koje se ubacuju sa palete *Delphi* obezbeđuje automatsko kreiranje i upis promena njenih propertija u **DFM** fajl). Da bi se nova komponenta ubacila ručno bez potrebe za menjanjem samih objekata, potrebno je editovati i **DFM** fajl i u njega ubaciti komponentu sa neophodnim vrednostima.

Suprotno **PAS** fajlovima, **DFM** fajlovi na disku postoje isključivo u binarnoj formi. Da biste mogli da ga editujete kao tekst fajl, morate ga učitati isključivo u *Delphi*-jev editor (**File|Open File**, pa izaberete **DFM** fajl) koji će ga iz binarne forme iskonvertovati u tekstualnu i omogućiti vam izmene. Pre nego što to i uradite (otvorite **DFM** fajl) snimite modul kako bi **DFM** fajl sadržavao vrednosti koje su menjane od zadnjeg snimanja. Ukoliko nete snimili modul, *Delphi* će vas upozoriti na to.

Struktura **DFM** fajla je veoma jednostavna. Sastoji se iz ugnježdenih (*nested*) struktura koje sadrže imena snimljenih propertija i njihove vrednosti (listing 5.3 sadrži tekst verziju **DFM** fajla *Hello World* aplikacije). Editovanje je vrlo prosto, samo morate paziti da se nove vrednosti propertija koje menjate slažu po tipu sa deklarisanim (dakle, ne možete stavljati stringove tamo gde idu brojevi). U slučaju da ste u **PAS** fajl ručno ubacili neku komponentu, potrebno je da ovde ubacite taj isti objekat sa spiskom propertija koje inicijalizujete. To će obezbediti da se vaša komponenta automatski kreira pri kreiranju formulara i inicijalizuje unetim vrednostima.

Listing 5.3

```
object Form1: TForm1
  Left = 200
  Top = 99
  Width = 435
  Height = 300
  Caption = 'Hello World Aplikacija'
  Font.Color = clWindowText
  Font.Height = -13
  Font.Name = 'System'
  Font.Style = []
  PixelsPerInch = 96
  TextHeight = 16
  object Label1: TLabel
    Left = 160
    Top = 104
    Width = 77
    Height = 16
    Caption = 'Hello World'
  end
  object OKButton: TButton
    Left = 15[illegible]
    Top = 1[illegible]
    Width = [illegible]
    [illegible]
    [illegible]
    [illegible]
    [illegible]
    [illegible]
  end
end
```

Pošto *Delphi* ne može da vodi računa o onome što unosite u tekst verziju **DFM** fajla, za vreme editovanja isključeni su sors fajl modula (**PAS**) i dizajn prozor istog. Nakon što završite editovanje **DFM** fajla i otvorite njegov **PAS** parnjak, ova dva prozora će se vratiti, a u dizajn prozoru će se registrovati sve promene koje ste ručno uneli u **DFM** fajl (ako ste na primer promenili naslov komponente). Probajte, na primer, da editujete **DFM** fajl *Hello World* aplikacije i izmenite *Caption* polje formulara tako što ćete dodati un znaka uzvika na postojeći string. Snimite **DFM** fajl i učitajte **PAS**. U dizajn prozoru trebalo bi da se automatski pojavi novi string.

To bi otprilike bio sistem kreiranja aplikacija. Ostaje vam da konsultujete tabelu komponenata na kraju ovog umetka i probate razne varijante.

Project Manager

Sve manipulacije vezane za sam projekat obavljaćete iz ovog prozora. Pored manje više standardnih komand (Add, Remove, View) najvažnija vam je komanda **Options**. Njome se otvara dijalog u kojem postavljate globalne opcije vezane za projekat. Neke od njih su jedinstvene (kao što je, na primer, *Windows* ime aplikacije) dok se neke mogu lokalno (na nivou sors fajla) promeniti, kao što je slučaj sa svičevima kompajlera.

Prva strana ovog dijaloga nosi naziv Formulari (*Forms*). U njemu se nalaze dve liste sa spiskovima formulara koje vaša aplikacija sadrži. Razlika između ove dve liste je što prva (*Auto-Create Forms*) sadrži formulare koji se automatski kreiraju na početku programa (njihova inicijalizacija se nalazi u **DPR** fajlu), dok se formulari sa druge liste (*Available Forms*) ne kreiraju automatski, već se ostavlja vama da ih kreirate. Za početak je najbolje ostaviti sve formulare u prvoj listi (gde se prvobitno smeštaju na početku dizajniranja) dok ne naučite da manipulišete istim. To što se svi formulari nalaze u prvoj listi ne znači da se svi odjednom pojavljuju na ekranu kad startujete program. Pojavljuje se samo jedan – to onaj čije se ime nalazi u polju sa nazivom *Main form* (odmah iznad liste). Ostali formulari se pri startovanju nalaze u memoriji aktivni ali skriveni. Po potrebi ih možete pozivati sa **<ime formulara> Show** (znači pozivanjem *Show* metode) i ponovo ih sakrivati sa metodom *Hide* (međusobna komunikacija više formulara).

Na drugoj strani (*Application*) možete postaviti *Windows* ime aplikacije (ono koje će se pojaviti ispod ikonice programa) kao i samu ikonicu (početna ikonica je standardna *Delphi*-jeva, znači hram sa plamenom). Takođe, postoji mogućnost da u *Help File* polje ubacite naziv standardnog *Help fajla* (ukoliko vaša aplikacija ima samo jedan).

Compiler stranica sadrži spisak kompajlerskih prekidača (switches) koji definišu način na koji će se kod generisati. Postoji nekih dvadesetak prekidača koji su namešteni na svoje početne vrednosti i koje možete menjati po želji.

Napomena: Ti stanja prekidača odnose se samo na sorsove koji nemaju u sebi definisane prekidače. Kompajler kada naiđe na neki prekidač privremeno (do kraja trenutnog fajla) menja vrednost prekidača. Po prelasku na novi fajl ponovo se vraćaju početne vrednosti definisane u ovom prozoru.

Na linker stranici prozora nalaze se opcije koje će vam rede trebati. Prvenstveno, kreiranje **MAP** fajla. Kada je ova opcija uključena na bilo koju vrednost sem *Off*, *Delphi*-jev linker će generisati odgovarajući tekst fajl (naziv fajla će odgovarati nazivu **DPR** fajla, samo će ekstenzija biti **MAP**) koji će sadržati odgovarajuće informacije o generisanoj aplikaciji. *Segments* deo će sadržavati spisak svih segmenata (poređanih adresno od imaginarnog početka memorije) sa dužinom, imenom i klasom. *Publics* deo sadrži spisak svih javnih (public) simbola (promenljive, procedure, ulazne tačke). *Detailed* predstavlja skup ovih informacija.

Link Buffer opcija daje mogućnost da odredite gde će se u procesu linkovanja smeštati informacije potrebne linkeru (identifikatori i njihove osobine). Preporučuje se korišćenje memorije (dobija se dosta na brzini), dok će *Disk* koristiti samo ako vam linker prijavi da nema dosta memorije (*out of memory*). Štaviše, preporučujemo da u takvim slučajevima povećate virtuelnu memoriju pre nego uključite ovu opciju, pošto će intern *Windows* ov swaper (menadžer virtuelne memorije) tokom linkovanja zadržati taj deo memorije u čopovima (zbog većeg broja pristupa), dok će *Delphi* dosta drijati po disku (naročito za veće projekte). **EXE** i **DLL** opcije (postoje dve) određuju da li će segmenti biti kompresovani i tako zauzimati manje mesta na disku, odnosno da li će se na **EXE** fajl nalikovati *Debug* informacije neophodne za debagovanje vaših programa iz *Turbo Debugera* za *Windows*. Za debagovanje programa iz samog *Delphi*-ja ova opcija nije neophodna.

Poslednja stranica prozora (*Directories/Conditionals*) predstavlja mesto gde možete navesti poziciju na kojoj će se kreirati **EXE** (**DLL**, **DCU** (*Delphi Compiled Unit*)) fajl. Podrazumeva se prazno polje, dakle završni fajl (ili modul) se kreira na mestu gde se nalazi odgovarajući sors fajl. Takođe, možete navesti direktorijume koje će *Delphi* pretraživati kad traži sors fajlove, kao i neke uslovne direktive (pored već definisanih).

OBJECT PASCAL

Najveća slabost većine trenutno postojećih *visual* alata je programski jezik u pozadini. Akcenat je u celosti prebačen na spoljašnji izgled i lakoću dizajna, dok je osnova na veoma klimavim nogama. Što za sobom vuče mnoge probleme. Prvi i možda najosnovniji je nemogućnost da se iskoristi sav potencijal operativnog sistema zbog apstrakcije, tj. pojednostavljivanja celog procesa razvoja aplikacije. Nemogućnost pravljenja izvršnih verzija je široko rasprostranjena "boljka" tih alata, najviše zbog velike zavisnosti aplikacije od razvojne okoline. Pokušaji da se od takvih aplikacija naprave **EXE** fajlovi najčešće preko (skupih) dodatnih *Development Kit*-ova, završavali su se dobijanjem glomaznih izvršnih fajlova i neophodnih, takođe glomaznih run time biblioteka. U nekim slučajevima distribucione verzije aplikacije dostizale je i par megabajta. Situacija se totalno promenila pojavom *Delphi*-ja. Na primer, jednostavan programčić tipa *Hello World* ima nešto ispod 200k, što je daleko manje od konkurentskih proizvoda.

U osnovi *Delphi*-ja nalazi se paskal kompajler sours-kompatibilan sa onim iz *Borland Paskala 7.0*, te je od njega preuzeo kako dobre, tako i loše osobine. Od ozbiljne optimizacije i dalje nema traga, isključujući case blokove pod određenim okolnostima. Uprkos tome, generisani izvršni fajlovi su kompaktni i brzi. Ma koliko interpreteri bili dobro napisani i optimizovani, njihova brzina daleko zaostaje za nezavisnim **EXE** fajlovima kakve generiše *Delphi*, što upravo predstavlja njegovu ključnu prednost nad ostalim **RAD** alatima. Jednostavno rečeno, korisnici vaših aplikacija neće morati da imaju instaliran *Delphi*, dovoljno je samo da startuju **EXE** fajl (izuzetak čine aplikacije koje koriste usluge **BDE**-a (*Borland Database Engine*) i *Report Smith*-a, kada se uz aplikaciju moraju isporučiti i distribution verzije ovih programa, koje možete naći na CD-u. Iako *Borland* preporučuje da se instaliraju u celosti, te distribution verzije mogu se znatno smanjiti, o čemu će biti reč u dodatku).

Podatak da sve komande koje postoje pod *Borland Pascal*-om 7.0 pod *Delphi*-jem generišu isti kod, navodi na zaključak da kompajler nije pisan ispočetka, već da je ubačen stari, proširen novim tehnikama. Upravo o tim proširenjima biće reči u ovom poglavlju, prvenstveno o novom objektnom modelu, pa preko obrade grešaka sve do otvorenih nizova. Tehnike koje su prenete iz **BP7.0** neće biti obrađene, pošto je prostor ovog umetka suviše mali za tako obimnu temu. Primeri koji su ovde navedeni u većini slučajeva neće moći da se prevedu, uglavnom zbog nekompletnosti. Cilj primera nije da proizvede neku akciju, već da prikaže kako bi to otprilike izgledalo.

NOVI OBJEKTNI MODEL

Borland Pascal 7.0 je sa svojim "starim" objektnim modelom predstavljao veliki problem za početnike, najviše zbog velikog posla oko definisanja, alokacije i kreiranja objekta. U pitanju su čisto šablonske tehnike, međutim sasvim dovoljne da potencijalnog korisnika odvuku od mnogo važnijih aspekata objektno orijentisanog programiranja, kao što su apstrakcija i polimorfizam.

Problem je rešen kompromisno, kreiranjem novog objektnog modela, umesto modifikovanja starog. Stari model je i dalje podržan, ali isključivo radi kompatibilnosti.

Postoji mnogo prednosti ovakvog razvoja stvari, između ostalog i potpuna kompatibilnost na source nivou, čega se *Borland* uvek pridržavao pri izbacivanju novih verzija. Ovaj podatak je veoma važan za programere koji prelaze sa **BP 7.0** na *Delphi*, a imaju već dosta urađenih stvari.

Nove klase se definišu rezervisanom rečju **class** sa sintaksom koja je identična staroj:

```
type TNoviObjekat = class (TObject)

{
polja i metodi objekta
}
end;
```

Postoji nekoliko funkcionalnih razlika između starog i novog modela.

Položaj instanci

U novom modelu instance se uvek nalaze na HEAP-u, tj. dinamički se alociraju, te se shodno tome u programima u principu isključivo pointeri na... Sam proces alokacije obavlja konstruktor, dok dealokaciju obavlja destruktor. Dakle, se još više pojednostavila upotreba, n... stano izbačena je upotreba kvalifikatora ^ pri radu sa pointerima koji se odnose na nove objekte. Na primer:

```
type TMyObj = class (TObject)
              public
                 polje : Word;
              end;

var inst : TMyObj;    {definisana prome-
ljiva je pointer na instancu}
begin
...
   inst.polje :=125;  {iako je inst po
inter na TMyObj ne koristi se ^ }
```

Konstruktori i Destruktori

Imena osnovnih konstruktora i destruktora promenjena su iz Init u *Create* (za konstruktore) i iz *Done* u *Destroy* (za destruktore). Druga promena je u tome što konstruktori imaju povratnu vrednost koja predstavlja pointer na instancu objekta. Ovo je bilo neophodno kad se uzme u obzir da je ceo proces alokacije instance prebačen na konstruktor. Dakle, kad kreirate instancu objekta morate zapamtiti pointer ukoliko imate nameru da manipulišete njime. Gornji primer bi u tom slučaju izgledao:

```
type TMyObj = class (TObject)
              public
                 polje : Word;
              end;

var inst : TMyObj;
begin
   inst := TMyObj.Create;   {kreiranje
instance tipa TMyObj}
   inst.polje :=125;
   inst.Destroy;            {dealocira-
nje instance objekta}
end.
```

Za razliku od starog objektnog modela, novi model dozvoljava virtuelne konstruktore. TObject.Create nije virtuelni metod, međutim većina ostalih konstruktora unutar **VCL**-a su virtuelni. Pošto se pri kreiranju instanci obavezno navodi tip objekta (kao u gornjem primeru), kompajler zna koji konstruktor treba da pozove bez obzira na to što on može biti virtuelan.

Novi specifikatori vidljivosti

Uvedena su dva nova specifikatora: protected i published. Prvi označava delove objekta kojima se može pristupati samo iz trenutnog i nižih objekata u hijerarhiji i najčešće se koristi za zaštitu kritičnih polja i metoda. Ova klauzula je dugo vremena postojala u C++ jeziku i bilo je krajnje vreme da se pojavi u *Object Pascal*-u. Ranije su programeri sva polja i metode koji treba da budu pristupačni naslednicima morali da deklarišu kao public, čime su ti podaci bili izloženi nekontrolisanom pristupu krajnjih korisnika. Pogrešno upisana vrednost u pogrešno polje lako je dovodila do neispravnosti rada objekta, a time i samog programa. Kod *Delphi*-jem se takva polja ili proglašavaju kao protected (zaštićena) čime se pristup ograničava preko rezervisane reči **property**.

Published je, sa druge strane, privilegija *Delphi*-ja i označava elemente kojima se može pristupiti iz samog *Delphi*-ja prilikom dizajniranja aplikacije. Svi podaci (*properties*) i događaji (*events*) koje vidite u *Object inspector*-u su u svojim objektima deklarisani u *published* sekciji. Prilikom izvršavanja programa svi published elementi se tretiraju kao public, tj. pristup im je neograničen (uzimajući propertije koji imaju svoj mehanizam zaštite).

Bazni objekat

Svi objekti potiču od jednog baznog objekta, **TObject**. Po tom pitanju nema mnogo izbora, pošto će sledeća deklaracija

```
type TMyObj = class
```

podrazumevati da se nasleđivanje vrši iz **TObject** klase. Logična posledica ovoga je da više nema baznih objekata (sem **TObject**), što i nije neki problem, dok je prednost postojanje osnovnih (*default*) konstruktora **Create** i destruktora **Destroy** koji obezbeđuju uspešnu alokaciju i dealokaciju instanci. Za potrebe polimorfizma moguće je kreirati kvazi bazni objekat izveden iz **TObject** apstraktnim metodama. Pomenuta "prinuda" neće uticati na ispravnost takve postavke objekata, čak naprotiv.

Properties

Properties polja su u stvari specijalne deklaracije koje saopštavaju kompajleru kako da ostvari pristup određenim poljima objekta. Pod ovim se podrazumeva čitanje, pisanje, default (početna) vrednost. Propertiji sami po sebi nisu polja objekta, ... status nekog (najčešće privatnog) polja ... Sintaksa glasi:

```
property <ime> [: tip]
   read <...>
   [ write <...> ]
   [ stored <...> ]
   [ default <vrednost> | nodefault ]
```

Na primer, ovako izgleda klasa objekta koji ima propertije:

```
type TMyObj = class
   private
      FPolje : Word;
   public
      property Polje : Word read FPolje
write FPolje default 123;
   end;
```

Dakle, postoji privatno polje **FPolje** (neobavezna konvencija nalaže da imena polja kojima će se pristupati preko property počinju sa F, slično konvenciji da se imena tipova pišu sa slovom T na početku) i jedan public (javni) property sa nazivom **Polje**. Ta linija označava da će se kvazi promenljiva **Polje** dobiti čitanjem i pisanjem u promenljivu **FPolje**, kao i da je osnovna (*default*) vrednost 123. Komanda dodeljivanja (a je tipa **TMyObj**)

```
a.Polje :=150;
```

...

```
a.FPolje :=150;
```

Razlog tome je što se u write delu definiše da upis ide u promenljivu **FPolje**, dakle bez ikakvih ograničenja. Ovakve property linije ređe ćete sretati, pošto od njih nema neke veće koristi. Tip propertija koristite samo za nove propertije, pošto je za redeklarisanje nekog skrivenog propertija dovoljno samo da napišete njegovo ime bez tipa. Ako pokušate da definišete novi property bez tipa, dobićete grešku.

Property se najčešće koriste u *published* sekciji klase da definiše polja koja će se pojaviti u *Object Inspector*-u, kontrolišući unos tokom dizajna. Neki property-ji su definisani kao *protected*, i da biste njih koristili *u Delphi*-ja, morate redeklarisati taj *property* bez tipa. Pored toga, može se koristiti i za kontrolu pristupa kritičnim poljima. To se postiže na dva načina.

1) Izostavljanjem *write* dela stavke. Taj potez će učiniti da se polje tretira kao *read only* (samo za čitanje). Upis u takav *property* neće biti dozvoljen iz jednog prostog razloga: kompajler neće znati kako da obavi dodelu.

2) Stavljanjem metode u *write* deo umesto imena polja. Kad god se posle piše u *property* automatski se poziva taj metod. Tako da ako na primer imamo sledeći objekat:

```
type TMyObj = class
   private
      FPolje: Word;
      procedure SetPolje(Value: Word);
   public
      property Polje: Word read FPolje
write SetPolje;
   end;
```

ako je instanca tipa **TMyObj** komanda

```
   instanca.Polje:=150
```

će se prevesti kao

```
   instanca.SetPolje(150);
```

na samom metodu je da prekontroliše unetu vrednost i upiše je (ili ne) u odgovarajuće polje. Konvencija nalaže da se metode koje se koriste u property-jima za čitanje i pisanje menjaju povezivanjem reči **Get** (čitanje) ili **Set** (pisanje) sa imenom property-ja. U gornjem primeru **SetPolje** označava ime metoda koji upisuje u *property* **Polje**.

Metod za upis (*write*) je obavezno procedura koja prima parametar istog tipa kao i *property*, dok je *read* metoda obavezno funkcija bez parametara koja vraća vrednost istog tipa kao i *property*. Ovo važi za sve tipove sem nizova čije metode imaju još parametara koji određuju indeks.

Store, *default*/*nodefault* stavke su veoma zanimljiva pojava u *Delphi*-ju, pošto se sve koriste isključivo zbog dizaniranja aplikacije. Kad unesete neku vrednost u *Object Inspector*, da bi se ona kasnije učitala, potrebno je da se snimi u DFM fajl. Sad, kad bi se sve vrednosti snimale, **DFM** fajl bi postao preglomazan, a sadržavao bi gomilu nepotrebnih podataka. Da bi se nekako regulisalo to snimanje, uvedene su ove tri rezervisane reči.

Store, koji ima za sebe logički izraz, određuje da li će *property* uopšte biti razmatran za snimanje. Pošto logički izraz ne mora biti samo *TRUE* ili *FALSE*, već može biti neko poređenje ili čak poziv neke logičke funkcije ili metoda, ova vrednost se izračunava prilikom snimanja DFM fajla, i ako je rezultat *FALSE*, vrednost ovog propertija biće ignorisana, dakle neće se snimati.

Ukoliko je rezultat logičkog izraza *TRUE*, konsultuje se *default*/*nodefault* sekcija. Ako je prisutno *default* <vrednost>, a vrednost propertija bude jednaka onoj u tom *default* delu (<vrednost>), *property* neće biti snimljen (pošto se podrazumeva da nije bio menjan). Ukoliko je bio naveden *nodefault*, vrednost će uvek biti snimljena, jer se podrazumeva da *property* nema početnu vrednost. Ne možete u isto vreme imati i *default* i *nodefault* rezervisane reči za jedan *property*, pošto su kontradiktorni.

Napomena. Ukoliko ne navedete neke od ovih rezervisanih reči podrazumeva se **STORE True NODEFAULT**, što znači da će takav *property* uvek biti snimljen u **DFM** fajl. Takođe je važno imati na umu da se osnovna vrednost u *default* delu ne dodeljuje automatski propertiju, već služi isključivo da *Delphi* odluči šta se snima, a šta ne. Na vama je da u konstruktoru nove komponente izvršite upis te iste *default* vrednosti u *property*. Ukoliko to ne uradite ili propertiju dodelite drugu vrednost iz konstruktora, imaćete u nekim slučajevima nedefinisane vrednosti.

Message hendleri

Iako klasna hijerarhija Delphi aplikacija prihvata i obrađuje većinu *Windows* poruka, ponekad se javlja potreba za ručnom obradom neke poruke (recimo, ne sviđa vam se kako neka komponenta obrađuje poruke o iscrtavanju ili želite da dodate obradu neke nestandardne poruke vašoj komponenti). Metode za obradu poruka definišu se rezervisanom reči u *message*, koju prati kod poruke. Svaka Windows poruka predstavljena je nekim brojem, ali je zbog lakoće upotrebe svim porukama (kodovima) dodeljena standardna konstanta. Npr. kod za poruku za iscrtavanje je 15 (0000000Fh), ali je za njega vezana konstanta **WM_PAINT**, koja pak više govori od broja. Hendleri za poruke primaju jedan parametar, obavezno deklarisan sa *VAR*. Primer:

```
   procedure Farbaj(var Info); message
WM_PAINT;
```


Hendleri poruka ne smeju imati ni jednu drugu direktivu sem *message*. Ime i tip parametra su nebitni i možete ih namestiti kako vam odgovara.

Direktive

U *Delphi* je ubačeno dosta direktiva koje olakšavaju posao. Pored već pomenute reči *message*, tu su: *dynamic*, *override*, *cdecl* i *abstract*.

Dinamički metodi su varijacija na temu virtuelnih. Obavljaju isti posao (koriste se za primenu polimorfizma) kao virtuelne metode, ali generišu manji kod (na uštrb brzine). U principu, virtuelne metode su efikasniji način primene polimorfizme (brže su), pa ćete dinamičke metode retko koristiti.

Da ne biste pri nasleđivanju razmišljali koja metoda je virtuelna, a koja dinamička, ubačena je direktiva *override*. Nju koristite kad menjate neku virtuelnu ili dinamičku metodu, i ona sama vodi računa o krajnjoj implementaciji. Važno je da *virtual* i *dynamic* reči koristite samo pri prvom deklarisanju metode, a za posle koristite *override*.

Pri pozivanju procedura i metoda pisanih u drugim jezicima (C, na primer) koristi se deklarisanje procedure sa dodavanjem direktive *cdecl* (*C DEClaration*). Kod ovakvih procedura i funkcija parametri se prenose na stek "naopako" i posle poziva stek se čisti iz modula koji poziva (C specifikacija). Pre je za pozivanje ovakvih procedura bilo potrebno napisati ih u C u kao *PASCAL* funkcije. **Napomena.** *Delphi* ne podržava pozivanje C funkcija sa promenljivim brojem parametara.

Abstract direktiva olakšava pisanje apstraktnih metoda. Pre je za isti posao metoda morala da se deklariše kao virtuelna, pa da se u telo metode ubaci poziv procedure *Abstract*. U *Delphi*-ju je stvar olakšana utoliko što kompajler automatski generiše telo funkcije sa tim pozivom za metode koje imaju ovu direktivu. *Abstract* se ne može koristiti za metode koje već nisu deklarisane kao *virtual* ili *dynamic*. Sve ostale osobine koje važe za apstraktne metode i dalje važe u *Delphi*-ju. Primer:

```
   procedure AbstraktnaMetoda; virtual; abstract;
```

Obrada grešaka

Pošto pojava greške u *Windows* programima često ima nezahvalan kraj (gubljenje resursa, blokiranje itd.) u *Delphi* je ubačen jedan (dve implementacije) mehanizam za zaštitu resursa i mehanizam za otkrivanje grešaka. Kad god u *Delphi* programu nastupi neka greška program generiše interni prekid. Mehanizam prepoznavanja i obrade tih prekida obavlja se preko dve strukture: **try finally end** i **try except end**.

Prva struktura koristi se isključivo za zaštitu resursa i funkcioniše na sledećem principu: ako se između *try* i *finally* javi neki prekid, izvršavanje automatski skače na deo između *finally* i *end*. U tom delu bi trebalo da se izvrši dealokacija zauzetih resursa. (Niko ne voli programe koji progutaju 20% resursa kad padnu). Ukoliko do greške nije došlo, izvršavanje prelazi na taj deo ... "po silu izvršavanja".

U drugoj strukturi stvari su malo drugačije. Cela svrha te strukture je u prepoznavanju različitih tipova greški i pisanju odgovarajućih prekidnih obrada koje će odraditi neki posao kako bi manje-više povratili program u stabilno stanje. Za razliku od prethodne strukture, po završetku rada zaštićenog bloka (između *try* i *except*) izvršavanje programa se nastavlja iza strukture.

Kad nastupi prekid, poziva se *finally*, odnosno *except* deo zadnjeg definisanog zaštićenog bloka. Blokovi se mogu nalaziti jedan unutar drugog, ali je to ugnježdavanje na nivou izvršavanja programa (ne kompajliranja). Ako se unutar zaštićenog bloka nalazi poziv procedure koja sadrži svoj zaštićeni blok, taj novi blok postaje aktuelan sve dok izvršavanje ne "izađe" iz njega.

Otvoreni nizovi

Jedna od lepših novina koju *Delphi* donosi sa sobom su otvoreni nizovi. Definišu se slično kao standardni nizovi, s tim što se izostavlja opseg niza, a primenjuju se za deklarisanje formalnih parametara funkcija i procedura. Primer:

```
procedure UradiNesto(const a: array of
Word);
```

Sam otvoreni niz referencira se kao klasični ... pomoć [ ]. Na primer, da biste pristupili petom elementu niza, jednostavno referencirate a[5] (u ovom primeru). Postoje tri funkcije uz pomoć kojih dobijate generalije niza:

- **Low()** – uvek vraća 0, tj. indeks prvog elementa
- **High()** – vraća indeks poslednjeg elementa ...

Ukupan broj elemenata je ova vrednost +1.

- **SizeOf()** – vraća veličinu niza (broj elemenata puta veličina jednog elementa).

Zgodna stvar sa ovim nizovima je to što nije neophodno deklarisanje niza i njegovo popunjavanje pre prosleđivanja procedure. Dovoljno je za pozivanje uraditi na primer:

```
UradiNesto([21,45,1,454,778]);
```

unutar gornje procedure za rezultate funkcije *High(a)* dobićemo 4, a za *SizeOf(a)* 10. Iako gornja struktura

podseća na skupove: tačno se zna koji element ide na koje mesto niza. Korisni primena ove tehnike može biti, na primer, u prosleđivanju koordinata nekog tela funkciji. Ne mora se kreirati nikakav **TRect** objekat, dovoljno je samo deklarisati funkciju kao otvoreni niz i pozivati funkciju kao što je to urađeno sa **UradiNesto**.

Rezultat funkcije

Kod prethodnih paskala primećen je jedan prilično nezgodan problem kod funkcija. Jedini način da se upiše rezultat funkcije je da se referencira ime funkcije. Sve to lepo funkcioniše dok vam ne zatreba upisana vrednost. Ako pokušate da pročitate rezultat preko imena funkcije (kao što ste u nju pisali), kompajler će to tretirati kao rekurzivni poziv.

Zato je uvađena kvazi promenljiva sa nazivom **result** koja će biti istog tipa kao povratna vrednost funkcije. Tu vrednost možete referencirati i za upis i za čitanje, bez straha od neželjene rekurzije. Na kraju funkcije ta vrednost će se automatski vratiti, pošto ova promenljiva ima adresu identičnu onoj na koju se upisuje vrednost pri referenciranju imena funkcije.

Promene u CASE strukturi

Veoma zanimljiva stvar dogodila se u Delphi-ju sa CASE strukturom. Pojavio se prvi tračak nade da će paskal jednog dana imati optimizaciju vrednu pomena. Ma kako to malo sad izgledalo, pak je bolje nego ništa. Ceo princip optimizacije zasniva se na tome da korisnik sam postavi konstante u pravilan raspored (da svaka sledeća bude veća od prethodne). U tom slučaju kompajler ne generiše nikakava izračunavanja već uvodi [illegible] kao niz ustrvenih skokova (što je dosta brže za veće CASE strukture). Druga izmena kod CASE blokova je to što više nisu dozvoljena preklapanja [illegible] ne sme biti:

```
case [illegible]
  1 [illegible]
  20 [illegible] 90 [illegible]
  50 [illegible] 5 [illegible]
end
```

inače ćete dobiti **"Duplicate CASE constant"** grešku.

DODATAK I

Meni: File



- **New Project** – zatvara trenutni projekat (ukoliko nije snimljen, Delphi će izdati upozorenje) i otvara novi projekat koji se sastoji iz samog projekta i jednog formulara (obično praznog). Naziv projekta biće **PROJECT1.DPR**, a formulara **UNIT1.PAS** (i **DFM**). Ova imena možete promeniti prilikom snimanja.
- **Open Project** – zatvara trenutni projekat (isto kao **New Project**) i otvara klasični **Open File** dijalog za učitavanje preko kojeg tražite (i konačno učitavate) **DPR** fajl. Automatski će se učitati i desktop fajl na osnovu čega Delphi otvara i raspoređuje prozore na ekranu u onom rasporedu kakav je bio pri snimanju.
- **Save Project** – snima projekat i sve formulare u njemu. Ukoliko fajlovi nisu ranije snimani, Delphi otvara za svaki fajl poseban Save As... dijalog u koji možete uneti nova imena (ili zadržati ona koje Delphi preporučuje). Po snimanju projekta automatski će se snimiti desktop fajl.
- **Close Project** – zatvara trenutni projekat, pošto ponudi snimanje istog (ukoliko postoje nesnimljene izmene).
- **New Form** – kreira (ubacuje u projekat) novi formular sa nazivom **UNITxx** gde je **xx** prvi slobodan broj. Zajedno sa dizajn prozorom otvara se i sors fajl modula. Formular će se automatski naći na listi create listi u opcijama projekta (**View|Project Manager|Options**) čime se obezbeđuje njegovo automatsko kreiranje.
- **New Unit** – za razliku od prethodne komande, ova kreira samo modul (bez formulara i dizajn prozora). U taj modul možete ubaciti razne rutine koje nisu vezane za određeni formular. Modul će automatski biti ubačen u uses sekciju projekta i kao takav biće održavan od strane Delphi-ja. **Napomena**: Ukoliko ovaj modul koristite iz drugih svojih modula i formulara, morate sami da ubacite njegovo ime u uses sekcije dotičnih modula. U tom slučaju Delphi ga neće održavati, pa ako dođe do neke promene u imenu, morate opet sami da menjate.
- **New Component** – izborom ove komande otvarate Component Expert dijalog koji u sebi sadrži tri polja. U prvo, Class Name, unosite ime vaše nove klase, dok u drugo polje, Ancestor Class, unosite ime već postojeće komponente koju nasleđujete. Na kraju izaberete stranicu palete na kojoj nova komponenta treba da se pojavi i pritisnete **Ok**. Delphi će otvoriti novi modul i "otkucati" okvir za razvoj nove komponente, tako da vama ostaje samo da ubacite izmene.
- **Open File** – Otvara Open File dijalog iz kojeg možete učitati modul ili neki drugi tekstualni fajl. Ukoliko modul ima zakačen formular, on se takođe otvara. Ova komanda je jedini način da se **DFM** fajl otvori i edituje u tekst obliku. Ova komanda se može dobiti i sa **F3**.
- **Save File** – Snima fajl na disk pod njegovim trenutnim imenom. Ukoliko fajl nije ranije sniman, otvara se Save As... dijalog u kojem možete izabrati ime i direktorijum pod kojim će fajl biti snimljen. Isti efekat (snimanje fajla) postiže se akceleratorom **F2**.
- **Add File to Project** – otvara Open File dijalog iz kojeg se može ubaciti fajl u projekat. Izabrani fajl se automatski ubacuje u uses sekciju **DPR** fajla i otvara na ekranu.
- **Remove File** – suprotno prethodnom, ova komanda otvara dijalog sa spiskom svih modula iz kojeg korisnik izabira koji modul treba izbaciti iz projekta. Izabrani modul se automatski uklanja iz uses sekcije **DPR** fajla. **Napomena**: Ukoliko ste izbačeni modul koristi [illegible]

drugih modula/formulara nakon izbacivanja samog modula ostaje na vama da očistite preostale module od svih referenci na obrisani modul. U suprotnom dobijaćete greške pri prevođenju.

• **Print** – štampa sadržaj trenutnog sorsa na izabrani štampač. Ukoliko ekstenzija fajla odgovara nekim od ekstenzija u **Options|Environment|Editor Options** biće ponuđeno štampanje sintaksnih razlika (kao ekvivalent bojama editora), odnosno u slučaju postojanja kolor štampača, štampanje u boji.

• **Exit** – izlaz iz *Delphi*-ja i kraj rada. Ukoliko su fajlovi menjani biće ponuđeno snimanje.

Meni: Edit



• **Cut** – iseca izabrani deo teksta i kopira ga u *Clipboard*. Kombinacija tastera za ovu komandu je **Shift+Del**.

• **Copy** – Samo kopira izabrani tekst u *Clipboard*, i ne izbacuje ga iz sorsa. Ova komanda se može dobiti i preko tastature kombinacijom **Ctrl+Ins**.

• **Paste** – Prebacuje tekst iz *Clipboard*-a u editor počev od trenutne pozicije kurzora. Kombinacija tastera je **Shift+Ins**.

• **Delete** – izbacuje izabrani deo teksta i ne kopira ga. Ukoliko greškom obrišete tekst, možete ga vratiti **Edit|Undo** komandom. Tastatura: **Ctrl+Del**.

Meni: Search



• **Find** – otvara dijalog preko kojeg obavljate pretraživanje trenutnog sors fajla. Pored polja za unos dela teksta po kojem se vrši pretraživanje, nalaze se i prekidači preko kojih definišete način pretraživanja (ignorisanje razlike između malih i velikih slova, traženje samo celih reči), kao i smer i oblast koja se pretražuje.

• **Replace** – otvara dijalog veoma sličan prethodnom, sa razlikom dodatnog polja gde unosite tekst kojim će se zameniti prvobitni.

• **Search Again** – ova komanda se nastavlja na **Find** komandu – pronalazi sledeće ponavljanje teksta koji je unesen u **Find** dijalog.

• **Browse Symbol** – otvara **Browser** i pozicionira ga na identifikator na kojem se kurzor nalazio (ukoliko je editor bio poslednji aktivni prozor), odnosno otvara dijalog za unos naziva identifikatora ukoliko reč ispod kurzora ne predstavlja identifikator.

Meni: View



• **Project Manager** – otvara *Project Manager* prozor iz kojeg se može vršiti manipulacija nad delovima ili celim projektom.

• **Project Source** – u *Delphi* editoru otvara sors fajl samog projekta. Projektni fajl je standardni paskal fajl, samo mu je ekstenzija **DPR**.

• **Object Inspector** – otvara *Object Inspector* prozor ukoliko ste ga greškom ili iz nekih drugih razloga zatvorili. Ova komanda se može dobiti sa **F11**.

• **Alignment palette** – prikazuje na ekranu prozor će sa ikonicama za poravnavanje elemenata. Koristite je tako što na dizajn prozoru izaberete dve ili više komponenata (držeći Shift), onda izaberete jedno dugme iz ovog prozora. U zavisnosti od ikonice na dugmetu izabrane komponente će se poređati u istu liniju.

• **Browser** – otvara **Browser** prozor.

• **Breakpoints** – otvara prozor sa spiskom svih definisanih prekidnih tačaka u projektu. Odavde možete editovati te prekide i dodavati im nove uslove. Nove prekidne tačke generišete dvostrukim klikom u prazan deo prozora ili izborom komande **Run|Add Breakpoint**. Linija koja sadrži prekid imaće drugačiju boju od ostalih (crvenu) i imaće znak **STOP** sertan ispred nje. Tačke prekida uklanjaju se pritiskom na **Del** u ovom prozoru ili pritiskom miša na znak **STOP** u editoru.

• **Call Stack** – otvara prozor u kojem se prate pozivi funkcija, procedure, metoda. Služi za praćenje toka programa prilikom debagiranja.

• **Watches** – otvara prozor sa listom svih promenljivih čije se stanje prati kroz program. Nove promenljive dodajete sa **Ins**, dok stare možete menjati sa **Enter**. Brisanje se obavlja tasterom **Del**.

• **Toggle Form/Unit** – ovom komandom se šetate između formulara (dizajn prozora) i njegovog sors fajla u editoru. Prebacivanje inače možete izvesti "vizuelno" izborom odgovarajućeg prozora. Međutim, ako se takvo prebacivanje ne može obaviti (jedan prozor prekriva drugi), koristi se ova komanda. Za ovu komandu postoji taster **F12**.

• **Units** – (**Select unit from list**) – otvara spisak svih modula (sa ili bez pridruženih formulara) u kojem birate koji ćete editovati. Po izboru, fokus se prebacuje na editor u koji se po potrebi (ukoliko već nije tu) učitava zahtevani modul. Ukoliko uz modul postoji formular i on se automatski učitava.

• **Forms...** (**Select form from list**) – otvara spisak svih formulara slično kao u prethodnoj komandi i po izboru prebacuje fokus na dizajn prozor (umesto na sors).

Meni: Compile



• **Compile** – ova komanda pokreće *Delphi*-jev prevodilac i prevodi sve fajlove iz projekta koji su menjani od zadnjeg prevođenja (uključujući DPR fajl koji se uvek prevodi iznova) a zatim generiše EXE fajl. Ova komanda neće snimiti fajlove koje trenutno menjate, ali će ih uzeti za kompajliranje. Na vama je da fajlove snimite. U slučaju da nema otvorenog projekta, prevodi se samo aktivni modul i ne generiše se EXE fajl. Ova komanda se može pozvati sa **Ctrl+F9**.

• **Build All** – ova komanda je identična prethodnoj s tim što se prevode apsolutno svi fajlovi koji su uključeni u projekat bez obzira na to da li su skoro menjani ili ne.

• **Syntax Check** – ova komanda prevodi sve module u projektu, ali ih ne linkuje u **EXE** fajl. Korisno ako samo testirate vaš program na sintaksne greške, a ne želite da se u slučaju uspešnog prevođenja bakćete uklanjanjem **EXE** fajla.

• **Information** – ova komanda otvara prozor u kojem su ispisane osnovne generalije poslednjeg prevedenog projekta/modula, kao što su veličina koda (*Code Size*), podataka (*Data*), steka (*Stack*) i lokalne dinamičke memorije (*Local Heap*) koja se alocira pre startovanja programa od strane *Windows*-a. Poslednje dve veličine možete menjati sami iz opcija projekta.

Meni: Run



• **Run** – ova komanda će rekompajlirati projekat (ako je nešto menjano od poslednjeg prevođenja) i startovati program. Isti efekat se postiže tasterom **F9**.

• **Step Over** – ovu komandu (koju je inače lakše koristiti sa **F8** tasterom) koristite tokom debag sesije kada ispitujete program. Njome izvršavate liniju po liniju koda i po potrebi pratite rezultate raznih promenljivih. Ova komanda, za razliku od sledeće, tretira pozive procedura, funkcija i metoda kao klasične komande i preleće preko njih (što joj i ime kazuje).

• **Trace Into** – komanda identična prethodnoj s tim što ne preleće preko poziva, već "ulazi" u pozvanu proceduru, funkciju ili metod i postavlja se na prvu liniju koda unutar nje. Ovu komandu ćete najčešće koristiti uz pomoć tastera **F7**.

• **Program Pause** – ova komanda će privremeno (do pritiska **F9** tastera, ili da kraja trenutnog bloka komandi) obustaviti rad programa. Ukoliko vam program uleti u neku mrtvu petlju dok radi pod *Delphi*-jem možete uz pomoć više uzastopnih pritsaka kombinacije **Ctrl-Alt-SysReq** zaustaviti program i otkloniti problem.

• **Show Execution Point** – ova komanda važi samo kad ste u debag sesiji. Postavlja kurzor na red u kojem se zaustavilo izvršavanje. Ukoliko dotični sors nije otvoren, *Delphi* će ga učitati.

• **Add Watch** – otvara dijalog preko kojeg ubacujete nove promenljive u *Watch* prozor (**View|Watches**). Ovaj prozor dobijate i kombinacijom tastera **Ctrl+F7**.

• **Add Breakpoint** – otvara dijalog preko kojeg ubacujete nove tačke prekida u listu istih. *Delphi* podrazumeva da hoćete da ubacite tačku prekida u trenutnu liniju editora (tamo gde se nalazi kurzor), tako da će većina polja biti popunjena tim podacima (sors fajl, broj linije).

• **Evaluate/Modify** – otvara istoimeni dijalog. Taj dijalog će sigurno biti poznat starijim korisnicima, pošto je efektna zamena za kalkulator i koristi se za pregledanje vrednosti određenih promenljivih u programu (**Evaluate**). Kao dodatak tome, moguće je i menjanje istih tokom izvršavanja programa (**Modify**).

• **Toggle Breakpoint** – Uključuje/isključuje tačku prekida na trenutnoj poziciji kurzora.

Meni: Options



• **Project** – otvara *Project Options* dijalog identičan onome koji se dobija komandom **Options** iz *Project Manager*-a.

• **Environment** – Otvara dijalog za konfigurisanje radne okoline. Više o ovom dijalogu bilo je u četvrtom poglavlju.

• **Tools** – otvara dijalog iz kojeg namještate stavke **Tools** menija (odmah pored **Options**). Stavke u tom meniju predstavljaju spoljne alatke koje *Delphi* standardno ne poziva i u potpunosti se mogu nameštati od strane korisnika. Ako vam je potrebno, u ovaj meni možete ubaciti *DOS Prompt*, muzički program, jednostavno sve što vam se učini potrebnim.

• **Rebuild Library** – uzimajući u obzir nove opcije postavljene u **Options|Environmnet|Library** *Delphi* prevodi sve module čije se komponente nalaze u biblioteci (ukoliko postoje njihovi sors fajlovi).

Meni: Help



• **Contents** – otvara *Help* prozor, učitava osnovni *Delphi*-jev Help fajl (u okviru *Delphi*-ja posto

i više Help fajlova) i prebacuje se na stranicu sa sadržajem.

• **Windows API** - pored Delphi-jevog helpa u paketu se nalazi (između ostalog) i help fajl sa spiskom svih Windows-ovih procedura, funkcija i poruka (*Windows API* - Application Programming Interface). *Većina tih funkcija i poruka već su uklopljene u hijerarhiju* **VCL**-a, međutim, može vam zatrebati da nešto uradite sami.

DODATAK II

Spisak komponenti

Stranica: Standard



• **MainMenu** - glavni meni. Svaki formular koji pretenduje da ima meni, mora imati jednu ovakvu komponentu. Važno polje mu je Items. Duplim klikom na to polje otvarate Meni dizajner preko kojeg možete vizuelno kreirati meni.

• **PopupMenu** - lokalni meni. Možete ga namestiti (postavljanjem **AutoPopup** na *True*) da se pojavljuje pritiskom na desno dugme. Elemente menija pravite na isti način kao i glavni meni.

• **Label** - statični tekst na ekranu. Može uz sebe imati vezanu jednu "fokusnu" komponentu (nju ubacujete u **FocusControl**) tako da pritisak na akcelerator definisan u labeli automatski prebacuje fokus na tu komponentu (sama labela ne može imati fokus). Sadržaj labele se ubacuje u **Caption** property.

• **Edit** - jednolinijsko polje za unos teksta. Sav tekst ukucan u ovoj komponenti odlazi u njen property **Text** odakle ga možete pročitati. Ukoliko je tekst menjan od poslednjeg čitanja, **Modified** property će biti *True*.

• **Memo** - slična **Edit** komponenti, s tom razlikom što **memo** komponenta može da edituje tekst u više linija.

• **Button** - klasično Windows dugme. U **Caption** unosite naslov, a na events strani namestite **OnClick** na metod koji će obavljati posao na pritisak dugmeta. **ModalResult** polje određuje da li će se formular zatvoriti izborom ovog dugmeta, dok **Default** i **Cancel** polja označavaju da li je dugme osnovno (može se aktivirati sa **Enter** bez fokusa) ili je dugme za odustajanje (aktivira se na **Esc** taster).

• **CheckBox** - ova komponenta predstavlja kontrolu koju ste često viđali u Windows-u. Pritiskom miša na nju uključujete, odnosno isključujete logičko polje **Checked**. Karakteristika ove komponente je da kada se nalazi u grupi sa više istih komponenata, možete nezavisno uključivati/isključivati bilo koju od njih.

• **RadioButton** - slično **CheckBox** komponenti, sa razlikom u grupisanju. Kada se više ovih kontrola nađe u jednoj grupi, samo jedna od njih može biti uključena. Uključivanje druge "stavke" automatski isključuje prethodno aktivnu. Razlika je takođe i u izgledu kontrole (kružići umesto kockica).

• **ListBox** - standardna Windows lista. Elemente liste ubacujete u **Items** a tokom izvršavanja programa uz pomoć metode **Add(const s:String)**. Lista može da se prostire na više kolona, što se određuje u propertiju **Columns**.

• **ComboBox** - ova kontrola predstavlja u pravom smislu reči kombinaciju **Edit** komponente i Windows liste. Sve što važi za te dve komponente, važi zbirno za ovu.

• **ScrollBar** - iako većina "rastegljivih" komponenti automatski kreira svoje **scrollbar**-ove, ponekad je potrebno ubaciti neku komandu. Kad god se promeni položaj kurzora unutar **scrollbar**-a, događa se **OnScroll** event. Unutar njega definišete akcije vezane za njega.

• **GroupBox** - standardna Windows grupa. Nakon što je postavite u formular, sve komponente koje postavite na nju biće **grupisane** u jednu funkcionalnu celinu. Najčešća primena ovog **grupisanja** je kod **RadioButton** komponenti.

• **RadioGroup** - grupa namenjena da se radio dugmići ne unose kao posebne komponente, već kao elementi liste (**Items** *property*). Samo kreiranje radio dugmića obavlja ova grupa.

• **Panel** - ova komponenta je čisto dizajnerska, ne obavlja nikakav posao, ali ako se pravilno namesti može u mnogome doprineti lepšem izgledu formulara.

Stranica: Additional



• **BitBtn** - prva poslastica sa ove stranice je dugme sa slićicom. Ima sve propertije kao i obično dugme sa dodatkom dva: Kind i Glyph. Dvostrukim klikom na Glyph dobijate dijalog iz kojeg možete učitati ikonicu. Neke od već definisanih ikonica možete dobiti izborom neke vrednosti u Kind polju. Tu se nalaze Ok, Cancel, Yes, No i druga dugmići.

• **SpeedButton** - ovi dugmići predstavljaju delove speedbar-a, identične onima koje ste videli u glavnom prozoru Delphi-ja. Najčešće se pravi jedan Panel i na njega redaju potrebni dugmići. Sam posao koji dugmić mora da obavi piše se u OnClick handleru, kao kod običnom dugmetu.

• **TabSet** - ova komponenta predstavlja ona ispupčenja na stranicama koje ste viđali po raznim dijalozima u Delphi-ju (na primer konfigurisanje radne okoline). Nezgodna stvar sa njom je što ne funkcioniše kad je sama, već je potrebno ugraditi jednu Notebook komponentu i povezati je sa ovim tabulatorom. Spisak stranica nalazi se u propertiju Tabs.

• **Notebook** - komponenta koja može da sadrži više stranica, od kojih prikazuje samo jednu. Kada je samostalna, ovoj komponenti se stranice "prebacuju" dodeljivanjem propertijima PageIndex i ActivePage. Najčešće se koristi uz TabSet komponentu, kako bi se korisniku omogućilo da ručno "prevrće" stranice.

• **TabbedNotebook** - kombinuje prethodne dve komponente, ali samo po principu rada. Izgled je drugačiji, pošto se tabovi nalaze iznad (a ne ispod, kao kod TabSet-a) stranice.

• **MaskEdit** - klasično Edit polje sa dodatom funkcionalnošću: moguće je u mask polje ubaciti "šiku" po kojoj mora da se ravna ulaz. Dvostrukim klikom na mask polje otvara se dijalog u kojem možete da unosite i na koja mesta testirate svoje "šike".

• **Outline** - komponenta za prezentaciju elemenata u stablu. Nove čvorove - elemente stabla dodajete metodima Add i AddObjects. Način na koji će se iscrtavati stablo određuje se u polju OutlineStyle.

• **StringGrid** - matrica stringova uz koje se mogu vezati instance objekata. Na ekranu se prikazuje u obliku klasične matrice, a može se iskoristiti da se izborom određenog polja iz tabele dobije neka instanca (vezana za taj string) koja bi se negde drugde primenila.

• **DrawGrid** - slično prethodnoj komponenti, ova predstavlja matricu elemenata uz koje su vezani objekti među tim elementima matrice više nisu stringovi, već slike.

• **Image** - prikazuje sliku koja se može ubaciti pri tokom dizajniranja, ili se može učitati tokom izvršavanja. Ključno polje je Picture, koje je tipa TPicture, objekat koji obuhvata sve tipove slika pod Windows-om.

• **Shape** - ovo je ne-fokusna komponenta čiji je jedini zadatak da iscrtava neki geometrijski oblik (krug, elipsa, kvadrat). Tip geometrijskog oblika bira se upisivanjem odgovarajuće vrednosti u property Shape.

• **Bevel** - ovim komponentama kreira se 3D izgled vaših formulara. Možete praviti ispupčenja, udubljenja, okvire, menjati im dimenzije i dubinu.

• **Header** - rastegljiva traka koja obično stoji na dnu prozora i predstavlja statusnu liniju. Header je podeljen na više delova, od kojih svaki ima svoj tekst podložan menjanju. Tekstovi koji se pojavljuju upisani su u property Sections.

• **ScrollBox** - kreira područje na formularu čija je radna površina mnogo veća od same komponente. Time se mogu dobiti formulari koji će imati fiksne delove (van scrollbox-a) i veliki radni deo unutar scrollbox-a.

Stranica: Data Access



• **DataSource** - izvor podataka za kontrole. Nijedna kontrola ne može primati podatke od tabele ili query-a. Potrebno je kreirati ovaj objekat i u njega ubaciti izvor (tabelu ili query).

• **Table** - Tabela (jedna DB datoteka) se ovom komponentom ubacuje u formular. Potrebno je namestiti iz koje baze se uzima tabela (DatabaseName), koje je ime tabele (TableName), i koji je aktivni indeks (IndexName). U ovoj komponenti mogu se namestiti one-to-many relacije postavljanjem MasterSource i MasterFields polja.

• **Query** - Slično tabeli, samo što se izbor tabele i filter za prosleđivanje slogova definiše u SQL-u.

• **StoredProc** - objekat koji služi za izvršavanje određenih snimljenih procedura na database serveru.

• **Database** - ovaj objekat vam nije neophodan, pošto ga Delphi generiše kreira kad god ima Table ili Query objekat. Ako ga sami napravite (Database), Delphi ga neće kreirati, a vi ćete imati veću kontrolu nad svim transakcijama u bazi.

• **BatchMove** - objekat koji vam omogućava prebacivanje i konvertovanje baza uz konvertovanje samih podataka (ako je potrebno).

• **Report** - ovaj objekt startuje ReportSmith RUNTIME i štampa unapred definisan izveštaj (koji ste kreirali u "regularnoj" verziji ReportSmith-a).

Stranica: DataControls



• **DBGrid** - Matrica za pregled baza, nalik onima iz Access-a i Quattro Pro: horizontalno se nalaze imena polja, dok su slogovi poređani po vertikali. Dovoljno je samo upisati ime DataSource-a (sva komunikacija između baze i formulara ide preko DataSource-a) u istoimeno polje.

• **DBNavigator** - prikazuje skup od sedam ikonica pomoću kojih se možete šetati po DataSource-u koji je vezan za njega. Polje VisibleButtons određuje koji dugmići će se pojaviti.

• **DBText** - ekvivalent Label komponente, sa tom razlikom što se Caption polje ne ubacuje ručno, već se čitava iz DataSource-a.

• **DBEdit** - ekvivalent Edit komponente za rad sa bazama. Da bi funkcionisala, ova komponenta mora imati postavljene DataSource i FieldName.

• **DBMemo** - uz pomoć ove komponente možete editovati i prikazivati text BLOB-ove (Binary Large OBjects) koji se nalaze u trenutnom slogu tabele.

• **DBImage** - služi za prikazivanje slika iz BLOB-a iz trenutnog sloga tabele.

• **DBListBox** - klasični listbox koji svoje podatke uzima iz tabele umesto iz klasičnih izvora.

• **DBComboBox** - zgodna opcija za izbor jednog polja iz tabele i njegovo editovanje, pošto se lista koja se poziva popunjava iz tabele.

• **DBCheckBox** - Nakon što se u poljima ValueChecked i ValueUnchecked nameste određene vrednosti (čija podudarnost sa stvarnim poljima daje ili True ili False), ovo polje se prikazuje ili uključeno ili isključeno.

• **DBRadioGroup** - Slično kao za DBCheckBox, samo što se radi o više polja (poređanih u Items propertiju), od kojih samo jedan može biti uključen i njegova vrednost se upisuje u polje baze.

• **DBLookupList** - obuhvata dva izvora podataka (DataSource) i omogućava da se kretanjem po jednoj bazi pregledavaju podaci iz druge.

• **DBLookupCombo** - isto kao DBLookupList, samo što se pregled ne vrši iz listbox-a, već iz kombinovane kontrole (edit+list).

Stranica: Dialogs



• **OpenDialog** - stavlja na raspolaganje Open File dijalog, iz kojeg možete pružiti korisniku šansu da izabere ime fajla koji se otvara. Napomena: Samo otvaranje i učitavanje fajla obavljate vi, ovaj dijalog koristite samo za pribavljanje imena. Dijalog se aktivira metodom Execute, a finalni rezultat (u slučaju da je povratna vrednost True) može se pročitati iz FileName propertija. U Options propertiju birate kako će izgledati i kako se ponašati sam dijalog.

• **SaveDialog** - Save File dijalog, skoro identičan prethodnom, uz razliku u nazivu dijaloga. Opet, samo snimanje fajla morate obaviti vi.

• **FontDialog** - nakon što u Device propertiju izaberete da li će promena fonta uticati na ekran, štampač ili oboje, startujete ovaj dijalog metodom Execute i izaberete odgovarajući font, veličinu, boju i tip slova. Ako je povratna vrednost True, u Font propertiju dijaloga nalazi se opis izabranog fonta.

• **ColorDialog** - metodom Execute poziva se dijalog iz kojeg korisnik može izabrati boju. Ako je povratna vrednost True, u Color propertiju možete naći izabranu boju.

• **PrintDialog** - metodom Execute pozivate ovaj dijalog iz kojeg korisnik može izabrati sve opcije vezane za štampanje, kao što su broj kopija, raspon stranica, štampanje u fajl, startovanje setup-a printera, itd. Samo štampanje morate obaviti sami.

• **PrinterSetupDialog** - poziva standardni setup prozor za konfigurisanje štampača, u kojem možete birati kvalitet štampe, veličinu papira, i sve ostale vrednosti koje podržava proizvođač drajvera.

• **FindDialog** - sa Execute se otvara dijalog za pretraživanje teksta. Kada korisnik unese tekst i izabere FindNext dugme, dijalog poziva OnFind događaj koji vaša aplikacija treba da preuzme i obavi pretraživanje.

• **ReplaceDialog** - isto kao FindDialog, samo što je dodat OnReplace događaj koji treba da izvrši izmene u tekstu.

Stranica: System



• **Timer** - ova komponenta vezana je za Windows ov timer i poziva OnTimer događaj u redovnim intervalima upisanim u polje Interval.

• **PaintBox** – omogućava iscrtavanje na delu formulara (ne na celoj površini) sa tačno definisanim granicama. Na vama je da preuzmete **OnPaint** događaj i da iscrtavate sadržaj **PaintBox** a preko njegovog **Canvas** objekta. Ukoliko želite da iscrtavate preko cele površine formulara, najbolje je da koristite **OnPaint** događaj samog formulara, umesto da pravite **PaintBox**.

• **FileListBox** – prikazuje listu fajlova u trenutnom direktorijumu.

• **DirectoryListBox** – prikazuje hijerarhijsku strukturu direktorijuma identičnu onima iz open/save file dijaloga.

• **DriveComboBox** – polje za izbor diska, kao ono iz open/save file dijaloga.

• **FilterComboBox** – polje iz kojeg se biraju filter (wildcards) koji će se prikazati negde. Ova komponenta i prethodne tri mogu se koristiti za pravljenje nekog "domaćeg" open/save dijaloga.

• **MediaPlayer** – specijalizovana komponenta koja se koristi za puštanje svih vrsta multimedijalnih fajlova za koje postoji instaliran drajver.

DODATAK III

Distribucija

Da biste distribuirali svoju aplikaciju krajnjim korisnicima, dovoljno je u većini slučajeva proslediti samo EXE fajl i DLL ove koje vaša aplikacija poziva. Nema nikakvih interpretera, runtime biblioteka, itd. Kada su u pitanju baze, stvari se malo menjaju. *Borland Database Engine* ne ugrađuje se u **EXE** fajl (kao što je bio slučaj sa *Paradox Engine*-om), već se instalira na nivou Windows-a. Iako cela distribuciona verzija **BDE** a zauzima dve diskete, nisu vam svi fajlovi neophodni. U tabeli **III** dat je spisak delova koje možete da iskopirate ako su vam neophodni, dok druge možete da ne brišete.

Napomena. Nemojte brisati fajlove na sistemu koji ima neke bazne aplikacije. Može vam se desiti da obrišete neki fajl koji je potreban toj drugoj aplikaciji.

Tabela III

Sve BDE aplikacije (ukupno 470k)	IDAPI01.DLL IDR10009.DLL ILD01.DLL IDAPI.CFG
SQL query	IDQRY01.DLL (mora uključivati i IDPDX01.DLL)
SQL i BatchMove	IDBAT01.DLL
Paradox baze	IDPDX01.DLL
B	IDDBAS01.DLL
ASCII tabele	IDASCI01.DLL
ODBC pristup	IDODBC01.DLL ODBC.NEW ODBCINST.NEW
BDE konfigurisanje	BDECFG.EXE BDECFG.HLP

SADRŽAJ

računari 113 ... čitalački servis

1 Gde upotrebljavate računar (izaberite više odgovora)

- ☐ na poslu
- ☐ kod kuće

2 Koje operativne sisteme koristite u radu (izaberite više odgovora)

- ☐ DOS
- ☐ Windows
- ☐ OS/2
- ☐ UNIX
- ☐ Mac System
- ☐ Netware

3 Koje su oblasti Vašeg interesovanja (izaberite više odgovora)

HARDVER

- ☐ Akvizicije podataka
- ☐ Besprekidno napajanje
- ☐ Dodaci
- ☐ Konfigurisanje
- ☐ Memorije / proširenja
- ☐ Miševi/grafičke table
- ☐ Modemske komunikacije
- ☐ Monitori
- ☐ Mreže
- ☐ Multimedija
- ☐ Prenosni računari
- ☐ Radne stanice
- ☐ Skeneri
- ☐ Skladištenje podataka
- ☐ Štampači / ploteri
- ☐ Zvuk

SOFTVER

- ☐ Baze podataka
- ☐ CAD/CAM
- ☐ Finansije
- ☐ Grafika
- ☐ Matematika / statistika
- ☐ Mreže
- ☐ Nauka
- ☐ Obrada teksta
- ☐ Obrazovanje
- ☐ Operativni sistemi
- ☐ Poslovanje
- ☐ Programski jezici/alati
- ☐ Tabelarni proračuni
- ☐ Zabava
- ☐ Zaštita podataka
- ☐ Zvuk

4 Kojom se vrstom posla bavite (izaberite jedan odgovor)

- ☐ Administracija
- ☐ Automatska obrada podataka
- ☐ Komercijala/marketing
- ☐ Rukovođenje
- ☐ Tehnička služba
- ☐ Učenik / student
- ☐ Ostalo ____________

5 Koja je Vaša uloga u procesu nabavke opreme u Vašem preduzeću (izaberite više odgovora)

- ☐ Inicijalno utvrđivanje potreba
- ☐ Izbor modela
- ☐ Izbor dobavljača
- ☐ Odobravanje
- ☐ Savetovanje

6 Koji su Vaši razlozi za upotrebu čitalačkog servisa (izaberite više odgovora)

- ☐ Nabavka u skorije vreme
- ☐ Nabavka u budućnosti
- ☐ Informisanje

moje savet

Zaokružite broj (najviše pet) za besplatne informacije

01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25

26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50

Ime i prezime ____________ God. rođenja ______

Ulica ____________ Broj ______

Mesto ____________ Poštanski broj ______

Pozivni broj ______ Telefon ______ E-mail ______

Naziv preduzeća ____________

NOVEMBAR '95 *Važi do 05 01 1996*

računari ... pretplata

Primeraka ______	6 meseci (6 brojeva)	12 meseci (11 brojeva)
1) za pravna lica (popust 10%)	☐	☐
2) za fizička lica (popust 20%)	☐	☐
3) za đake i studente (popust 30%)	☐	☐

Ime i prezime ____________

Ulica ____________

Mesto ____________

Naziv preduzeća (samo za pravna lica) ____________

Pretplatnik se obavezuje da će izvršiti uplatu najkasnije 7 dana po prispeću ustalnice. Potvrdu o ... faksom. Za đake i studente obavezno je priložiti potvrdu škole / fakulteta.

YUEDI ... prijavni obrazac

Ovim se neopozivo prijavljujem za učešće na 3 YUEDI konferenciji sa temom Elektronska razmena podataka i elektronska trgovina u svetu i Jugoslaviji stanje i perspektive koja će se održati u Beogradu 12 i 13 decembra 1995 god u Privrednoj komori Jugoslavije

IME I PREZIME ____________

PREDUZEĆE ____________

ADRESA STANOVANJA ____________

ADRESA PREDUZEĆA ____________

TELEFON ______ TELEFAKS ______

ELEKTRONSKA ADRESA ______

U ______

Datum ______ M P Potpis ______

Kotizacija za učešće na 3 YUEDI konferenciji iznosi 250,00 din za članove a 300,00 din za ostale ukoliko uplatu izvrše do 25 novembra 1995 g 300,00 din za članove a 350,00 din za ostale ako uplatu izvrše posle 25 novembra 1995 g a najkasnije do 10 decembra 1995 g Posle tog roka cene su 350,00 din za članove a 400,00 din za ostale Uplata se vrši na žiro račun YUEDI asocijacije br 40802-678-0-101808 sa napomenom "3 YUEDI konferencija"

POŠTARINA ZA TERITORIJU SRJ
PLAĆENA PO UGOVORU

Personalni računari

pretplata

BIGZ
Bulevar Vojvode Mišića 17
11000 Beograd

YUEDI

YUEDI ASOCIJACIJA

prijavni obrazac

(3. YUEDI konferencija)
Kneza Miloša 20
YU-11000 Beograd

POŠTARINA ZA TERITORIJU SRJ
PLAĆENA PO UGOVORU

Personalni računari

čitalački servis

BIGZ
Bulevar Vojvode Mišića 17
11000 Beograd